

明 細 書

キャッシュメモリおよびその制御方法

技術分野

本発明は、プロセッサのメモリアクセスを高速化するためのキャッシュメモリおよびその制御方法に関する。

背景技術

キャッシュメモリは、主記憶装置のアクセス時間を短縮しプロセッサの処理能力の向上を図るために、従来から広く用いられている。

例えば特開平 6 - 2 6 6 6 2 0 号公報等の開示されたキャッシュメモリは、メインメモリのブロック単位のデータを各エントリに格納し、このエントリを介して、マルチタスク処理を行なう処理ユニットからのアクセスに対応するデータの転送制御と排他制御を行なう。このキャッシュメモリは、エントリに格納されているブロックを排他制御の対象として設定した処理ユニットの各タスクの識別情報を登録するタスク識別情報登録部を、エントリごとに設ける構成とし、タスク単位で、エントリに格納されているブロックの排他制御、および、この排他制御の設定と解除を行なっている。

このキャッシュメモリによれば、マルチタスク処理における排他制御を効率良く行ない、タスク間で共通に使用するデータの矛盾を解消することを図っている。

しかしながら、上記従来技術におけるキャッシュメモリによれば、プロセッサのタスク切り替えに伴ってキャッシュメモリのヒット率が実行中でない他のタスクによる影響を受けるという問

題がある。

例えば、タスク A の命令列（又はデータ）がキャッシュメモリに格納されている状態でタスク A の実行からタスク B の実行に切り換えられた場合、タスク B の実行によりキャッシュメモリ中のタスク A の命令列（又はデータ）が追い出されてしまう。タスク A の命令列（又はデータ）がキャッシュメモリから追い出されていれば、再度タスク A が実行されたときに、キャッシュミスが発生するという問題がある。特に、圧縮音声データや圧縮映像データのデコード／エンコード処理などのリアルタイム性を必要とする処理では、上記タスク切り替えに伴う他のタスクの影響によって、タスク切り替え後のキャッシュのリプレース処理によってタスクの割当時間を侵食され、必要な処理時間を確保できず、リアルタイム性が損なわれるあるいは処理時間を確定できないという問題がある。

15

発明の開示

本発明は、タスク切り替え等によるキャッシュメモリの他のタスクの影響を防止し、タスクの実質的な処理時間を容易に確保するキャッシュメモリを提供することを目的とする。

20 上記目的を達成するため本発明のキャッシュメモリは、N－ウェイ・セット・アソシエイティブ方式のキャッシュメモリであって、N 個のウェイのうち 1 つ以上のウェイを示す制御レジスタと、制御レジスタに示されるウェイをアクティブにする制御手段と、制御レジスタの内容を更新する更新手段とを備える。

25 この構成によれば、キャッシュメモリを構成する N－ウェイのうち、制御レジスタに示されたウェイのみをアクティブにし、し

かも制御レジスタの内容は更新可能なので、プロセッサが実行する処理に応じてアクティブなウェイを動的に設定することができる。タスクとウェイとを対応付ければ、タスク切り替え後に他のタスクによりキャッシュメモリから必要なデータが追い出されることが解消され、タスク切り替えに伴うヒット率の他のタスクからの影響を防止することができる。その結果、タスクに必要とされる実質的な処理時間を容易に確保することができる。

ここで、前記制御手段は、制御レジスタに示されたアクティブなウェイ以外のウェイ（インアクティブなウェイと呼ぶ。）に対して少なくともリプレースを制限する構成としてもよい。

この構成によれば、インアクティブなウェイについては、少なくともリプレースが制限される。つまり、インアクティブなウェイについて完全にディスエーブルにしても、リプレースだけをディスエーブルにしてもよい。後者の場合、キャッシュメモリに対するリード／ライトまでは制限されないので、ヒット率の低化を防止し、かつインアクティブなウェイを有効に活用することができる。

ここで、前記キャッシュメモリは、さらに、ウェイ毎に設けられ、キャッシュデータのアドレスをタグとして保持するタグ保持手段と、プロセッサから出力されるメモリアクセスアドレスの上位部分であるタグアドレスと、タグ保持手段から出力されるN個のタグとを比較することによりヒットかミスヒットかを判定するN個の比較手段を有し、前記制御手段は、制御レジスタに示されたアクティブなウェイ以外のウェイに対応する比較手段をディスエーブルにする構成としてもよい。

この構成によれば、インアクティブなウェイに対応する比較手

段をディスエーブルにするので、比較手段における消費電力を低減することができる。

ここで、前記制御手段は、さらに、制御レジスタに示されたアクティブなウェイ以外のウェイに対応するタグ保持手段に対して、比較手段へのタグ出力をディスエーブルにする構成としてもよい。

この構成によれば、インアクティブなウェイに対応するタグ出力と比較手段とがディスエーブルされるので、タグ保持手段の消費電力を低減することができる。

10 ここで、前記制御手段は、プロセッサからメモリアクセスアドレスが出力されたとき、当該アクセスアドレスについて、比較手段に最大2回のタグ比較を行わせるよう制御し、1回目のタグ比較では、制御レジスタに示されたアクティブなウェイ以外のウェイに対応する比較手段をディスエーブルし、1回目のタグ比較においてミスヒットと判定された場合に、アクティブなウェイ以外のウェイに対応する比較手段をディスエーブルしないで2回目のタグ比較を行わせる構成としてもよい。

この構成によれば、1回目のタグ比較におけるヒット率が高いほど、比較手段における消費電力を低減することができ、しかも、
20 1回目のタグ比較においてミスヒットした場合に2回目のタグ比較を行うので、インアクティブなウェイのキャッシュデータも有効に活用することができる。

ここで、前記制御手段は、前記2回目のタグ比較においてアクティブなウェイに対応する比較手段をディスエーブルする構成
25 としてもよい。

この構成によれば、2回目のタグ比較ではインアクティブなウ

エイに対応する比較手段のみがタグ比較を行うので、さらに消費電力を低減することができる。

ここで、前記制御手段は、制御レジスタに示されたアクティブなウェイ以外のウェイに対して、その状態の更新を禁止する構成
5 としてもよい。

この構成によれば、例えばインアクティブなウェイの状態を示すフラグ類の更新を禁止することにより、インアクティブなウェイに対するタスク切り替えによる影響を防止することができる。

ここで、前記制御手段は、制御レジスタに示されたアクティブ
10 なウェイ以外のウェイについて、そのアクセス順序を示す情報の更新を禁止する構成としてもよい。

この構成によれば、アクセス順序を示す情報の更新を禁止するので、更新手段による更新によってインアクティブなウェイがアクティブなウェイに割り当てられたタスクのキャッシュミス時の
15 リプレース順が変化することがない。

ここで、前記キャッシュメモリは、さらに、前記更新手段によって制御レジスタの内容が更新されたとき、ウェイに対するアクセス順序を示す情報をリセットするリセット手段を有する構成としてもよい。また、前記アクセス順序を示す情報は、キャッシュ
20 エントリー毎の1ビットデータであり、前記キャッシュメモリは、さらに、リプレース可能な複数ウェイから1つのウェイをラウンドロビン方式で選択するためラウンド位置を示すデータを保持するレジスタを有し、前記リセット手段は、前記更新手段によって制御レジスタの内容が更新されたとき、前記レジスタをリ
25 セットするように構成してもよい。

この構成によれば、インアクティブなウェイを割り当てたタス

クのキャッシュミス時のリプレース順への影響をなくすることができる。

ここで、前記更新手段は、アクティブにすべきウェイを指定するウェイデータであって、タスク毎のウェイデータを保持する保持手段と、実行中のタスクに対応するウェイデータを保持するよう前記制御レジスタを書き換える書き換え手段とを有する構成としてもよい。

この構成によれば、タスクが切り替わる毎に動的に、制御レジスタを書き換えるので、他タスク毎にアクティブなウェイを対応付けることができる。

ここで、前記保持手段は、メモリ中に記憶されたタスク毎のコンテキストデータの一部として前記ウェイデータを保持し、前記書き換え手段は、タスク切り替えに際して、制御レジスタ中の現タスクのウェイデータをメモリに退避し、次タスクのウェイデータをメモリから前記制御レジスタに復帰する構成としてもよい。

この構成によれば、制御レジスタの更新は、OS（Operating System）によるタスク切り替えにより、キャッシュメモリのハードウェアを大幅に追加することなく簡単に実現することができる。

ここで、前記保持手段は、タスク毎の前記ウェイデータを保持し、前記書き換え手段は、メモリに記憶された各タスクのアドレス範囲を記憶するアドレス記憶手段と、アドレス記憶手段に記憶されたアドレス範囲と、プロセッサから出力される命令フェッチアドレスとに基づいて、実行中のタスクを判別する判別手段と、判別された実行中のタスクに対応するウェイデータを前記保持手段から選択する選択手段と、選択されたウェイデータを前記制

御レジスタに書き込む書き込み手段とを備える構成としてもよい。

この構成によれば、制御レジスタの更新は、キャッシュメモリ自身が主体的に判断することにより行われるので、どのようなプロセッサに対しても、タスク毎に対応するウェイをアクティブに
5 することができる。

ここで、前記保持手段は、タスク毎の前記ウェイデータを保持し、前記書き換え手段は、プロセッサから出力されるタスク番号に従って、実行中のタスクに対応するウェイデータを前記保持手段
10 段から選択する選択手段と、選択されたウェイデータを前記制御レジスタに書き込む書き込み手段とを備える構成としてもよい。

この構成によれば、プロセッサから出力されるタスク番号を利用するのでハードウェアを大幅に追加することなく、制御レジスタを簡単に更新し、タスク毎に対応するウェイをアクティブに
15 することができる。

ここで、前記保持手段に保持されるウェイデータは、OSによってタスクに割り当てられるように構成してもよい。

この構成によれば、タスクへのウェイの割り当てをOSが行うことによって、各タスクへのウェイの割り当てを最適化することが
20 容易になる。

ここで、前記キャッシュメモリは、各ウェイにおけるリプレース単位をキャッシュエントリーのラインサイズと、ラインサイズの2の n 乗分の1のサイズとに切り替え可能であり、前記制御レジスタは、さらに、ウェイ毎のリプレースサイズを示し、前記制
25 御手段は、制御レジスタに示されたリプレースサイズを単位としてリプレース制御を行う構成としてもよい。

また、前記制御手段は、制御レジスタに示されたアクティブなウェイ以外のウェイに対して少なくともリプレースを制限し、制御レジスタに示されたアクティブなウェイに対して制御レジスタに示されたサイズを単位にリプレースを行う構成としてもよい。

ここで、前記更新手段は、アクティブにすべきウェイを指定するウェイデータであってタスク毎のウェイデータと、タスク毎のリプレースサイズとを保持する保持手段と、実行中のタスクに対応するウェイデータ及びリプレースサイズを保持するよう前記制御レジスタを書き換える書き換え手段とを有する構成としてもよい。

この構成によれば、タスク毎にアクティブなウェイを切り換えると同時に、リプレース単位をも切り換えることができるので、タスクの処理内容に応じてミスヒットを低減することができる。

また、本発明のキャッシュメモリの制御方法は、N-ウェイ・セット・アソシエイティブ方式のキャッシュメモリを制御する制御方法であって、N個のウェイのうち1つ以上のウェイを示すウェイデータを制御レジスタに設定するステップと、制御レジスタに示されるウェイをアクティブにする制御ステップとを有する。

ここで、前記制御ステップでは、制御レジスタに示されたアクティブなウェイ以外のウェイに対して少なくともリプレースを制限するようにしてもよい。

ここで、前記制御方法は、さらに、アクティブにすべきウェイを指定するウェイデータであってタスク毎のウェイデータを保持する保持部から、実行中のタスクに対応するウェイデータ読み出して、読み出したウェイデータを前記制御レジスタに書き込む

更新ステップを有する構成としてもよい。

以上説明したように、本発明のキャッシュメモリによれば、プロセッサが実行する処理毎にアクティブなウェイを動的に設定することができるので、タスクとウェイとを対応付ければ、タスク切り替え後に他のタスクによりキャッシュメモリから必要なデータが追い出ることが解消され、タスク切り替えに伴うヒット率の他のタスクからの影響を防止することができる。その結果、タスクに必要とされる実質的な処理時間を容易に確保することができる。

10

図面の簡単な説明

図 1 は、本発明の実施の形態 1 におけるプロセッサ、キャッシュメモリ、メモリを含むシステムの概略構成を示すブロック図である。

15 図 2 は、キャッシュメモリの構成例を示すブロック図である。

図 3 は、ウェイ・レジスタのビット構成を示す図である。

図 4 は、ウェイ・レジスタとウェイとの対応関係を示す説明図である。

20 図 5 は、制御部におけるリプレース処理を示すフローチャートである。

図 6 は、タスク切り替え処理を示すフローチャートである。

図 7 は、タスク領域のコンテキストに含まれるタスク毎のウェイデータを示す図である。

25 図 8 は、本発明の実施の形態 2 におけるキャッシュメモリの構成を示すブロック図である。

図 9 は、キャッシュエントリーのビット構成を示す図である。

図 1 0 は、設定部の構成を示すブロック図である。

図 1 1 は、フラグの更新例を示す説明図である。

図 1 2 は、フラグ更新処理フローを示す図である。

図 1 3 は、リプレイス処理フローを示す図である。

5 図 1 4 は、本発明の実施の形態 3 におけるキャッシュメモリの構成を示すブロック図である。

図 1 5 は、キャッシュエントリーのビット構成を示す図である。

図 1 6 は、リプレイスサイズレジスタのビット構成を示す図である。

10 図 1 7 は、1 2 ベース フラグの更新例を示す説明図である。

図 1 8 は、ベース リプレイス処理を示すフローチャートである。

図 1 9 は、本発明の実施の形態 4 における比較制御部および各ウェイの要部の構成を示すブロック図である。

15 図 2 0 は、イネーブル回路の制御論理を示す真理値表をである。

発明を実施するための最良の形態

(実施の形態 1)

図 1 は、本発明の実施の形態 1 におけるプロセッサ 1、キャッシュメモリ 3、メモリ 2 を含むシステムの概略構成を示すブロック図である。同図のように、本発明のキャッシュメモリ 3 は、プロセッサ 1 およびメモリ 2 を有するシステムに備えられる。プロセッサ 1 は、マルチタスク制御を行うプロセッサであり、メモリ 2 中のタスク 1 ~ 4 等を切り替えて実行する。キャッシュメモリ 3 は、N-ウェイ・セット・アソシエイティブ方式のキャッシュメモリであって、タスク毎に N 個のウェイのうち 1 つ以上のウェ

20

25

イを対応させて、当該タスク実行中に対応するウェイをアクティブにするよう構成されている。各タスクは、アクティブなウェイに対しては、キャッシュメモリとしての全機能を利用可能である。また、各タスクは、アクティブでないウェイ（インアクティブな
5 ウェイと呼ぶ）に対しては、キャッシュメモリとしての全機能のうち利用可能な機能が制限されている。本実施の形態では、インアクティブなウェイは、リプレースする機能が制限され、それ以外のリード、ライト等は制限されていないものとする。

以下では、キャッシュメモリ 3 の具体例として、8 ウェイ・セット・アソシエイティブ方式のキャッシュメモリに本発明を適用
10 した場合の構成について説明する。

図 2 は、キャッシュメモリ 3 の構成例を示すブロック図である。同図のように、キャッシュメモリ 3 は、アドレスレジスタ 20、デコーダ 30、8 つのウェイ 31 a ~ 31 h（以下ウェイ 0 ~ 7
15 と略す）、8 つの比較器 32 a ~ 32 h、8 つのアンド回路 33 a ~ 33 h、オア回路 34、セクタ 35、セクタ 36、デマルチプレクサ 37、制御部 38 を備える。

アドレスレジスタ 20 は、メモリ 2 へのアクセスアドレスを保持するレジスタである。このアクセスアドレスは 32 ビットであるものとする。同図に示すように、アクセスアドレスは、最上位
20 ビットから順に、21 ビットのタグアドレス、4 ビットのセットインデックス（図中の S I）、5 ビットのワードインデックス（図中の W I）を含む。

ここで、タグアドレスはウェイにマッピングされるメモリ中の
25 領域（そのサイズはセット数×ブロックである。）を指す。この領域のサイズは、タグアドレスよりも下位のアドレスビット（A

10～A0)で定まるサイズつまり2kバイトであり、1つのウェイのサイズでもある。セットインデックス(SI)はウェイ0～3に跨る複数セットの1つを指す。このセット数は、セットインデックスが4ビットなので16セットである。タグアドレスおよびセットインデックスで特定されるブロックは、リプレース単位であり、キャッシュメモリに格納されている場合はラインデータ又はラインと呼ばれる。ラインデータのサイズは、セットインデックスよりも下位のアドレスビットで定まるサイズつまり128バイトである。1ワードを4バイトとすると、1ラインデータは32ワードである。ワードインデックス(WI)は、ラインデータを構成する複数ワード中の1ワードを指す。アドレスレジスタ20中の最下位2ビット(A1、A0)は、ワードアクセス時には無視される。

デコーダ30は、セットインデックスの4ビットをデコードし、8つのウェイ0～7の同順に位置するキャッシュエントリーからなる16個のセット中の1つのセットを選択する。

ウェイ0～7は、同じ構成を有数する8つのウェイであり、8×2kバイトの容量を有する。ウェイ0は、16個のキャッシュエントリーを有する。1つのキャッシュエントリーは、バリッドフラグV、21ビットのタグ、128バイトのラインデータを保持する。バリッドフラグVは、そのキャッシュエントリーが有効か否かを示す。タグは21ビットのタグアドレスのコピーである。ラインデータは、タグアドレスおよびセットインデックスにより特定されるブロック中の128バイトデータのコピーである。また、ウェイ1～7についても、ウェイ0と同様である。セットインデックスの4ビットによってデコーダ30を介して選択され

る 4 ウェイに跨る 4 つのキャッシュエントリーは、セットと呼ばれる。また、同図では書き込みがあったことを示すダーティフラグは省略されている。

5 比較器 3 2 a は、アドレスレジスタ 2 0 中のタグアドレスと、セットインデックスにより選択されたセットに含まれる 4 つのタグ中のウェイ 0 のタグとが一致するか否かを比較する。比較器 3 2 b ~ 3 2 h についても、ウェイ 3 1 b ~ 3 1 h に対応すること以外は同様である。

10 アンド回路 3 3 a は、バリッドフラグと比較器 3 2 a の比較結果とが一致するか否かを比較する。この比較結果を h 0 とする。比較結果 h 0 が 1 である場合は、アドレスレジスタ 2 0 中のタグアドレスおよびセットインデックスに対応するラインデータが存在すること、つまりウェイ 0 においてヒットしたことを意味する。比較結果 h 0 が 0 である場合は、ミスヒットしたことを意味する。15 アンド回路 3 3 b ~ 3 3 h についても、ウェイ 3 1 b ~ 3 1 h に対応すること以外は同様である。その比較結果 h 1 ~ h 7 は、ウェイ 1 ~ 7 でヒットしたかミスしたかを意味する。

20 オア回路 3 4 は、比較結果 h 0 ~ h 3 のオアをとる。このオアの結果を h i t とする。h i t は、キャッシュメモリにヒットしたか否かを示す。

セクタ 3 5 は、選択されたセットにおけるウェイ 0 ~ 7 のラインデータのうち、ヒットしたウェイのラインデータを選択する。

25 セクタ 3 6 は、セクタ 3 5 により選択された 3 2 ワードのラインデータのうち、ワードインデックスに示される 1 ワードを選択する。

デマルチプレクサ 3 7 は、キャッシュエントリーにデータを書

き込む際に、ウェイ 0～7 の 1 つに書き込みデータを出力する。
この書き込みデータはワード単位でよい。

制御部 38 は、内部にウェイ・レジスタ 371 を有し、キャッシュメモリ 3 の全体の制御を行う。ウェイ・レジスタ 371 は、
5 ウェイ 0～7 のうちアクティブなウェイを示すデータを保持するレジスタである。制御部 38 は、ウェイ・レジスタ 371 によって示されるアクティブなウェイに対しては、キャッシュメモリとしての全機能を制限なく制御し、インアクティブなウェイに対しては、リプレースする機能を制限する。

10 図 3 は、ウェイ・レジスタ 371 のビット構成を示す図である。同図のように、ウェイ・レジスタ 371 は、32 ビットレジスタであり、下位 8 ビットにウェイ 0～7 に対応する W0 フラグ～W7 フラグを保持する。例えば、W0 フラグが 1 のときウェイ 0 がアクティブなウェイであることを示し、0 のときウェイ 0 がイン
15 アクティブなウェイであることを示す。W1 フラグ～W7 フラグについても同様である。以下、W0 フラグ～W7 フラグの集まりをアクティブウェイデータと呼ぶ。このウェイ・レジスタ 371 は、プロセッサ 1 から直接読み書き可能であり、各タスクのコンテキストの一部をなす。つまり、タスク毎にアクティブウェイデ
20 ータを有し、タスク切り替えによって、ウェイ・レジスタ 371 の内容は実行中のタスクに対応するアクティブウェイデータに書き換えられる。

図 4 は、ウェイ・レジスタ 371 とウェイとの対応関係を示す説明図である。同図左側では、ウェイ・レジスタ 371 に保持さ
25 れているアクティブウェイデータが”00111000”であるので、ウェイ 2、3、4 がアクティブウェイとなり、ウェイ 0、

1、5、6、7がインアクティブになる。タスク切り替えに際して、ウェイ・レジスタ371は、例えば同図右側のようなアクティブウェイデータに書き換えられる。同図右側では、ウェイ5～7がアクティブとなり、ウェイ0～4がインアクティブになる。

5 図5は、制御部38におけるリプレース処理を示すフローチャートである。同図において、制御部38は、ミスヒットが発生したか否かを判定し（S51）、ミスヒットが発生したと判定された場合に、セットインデックスにより選択されたセットにおける、4つウェイのキャッシュエントリーの中からリプレース対象を
10 1つ選択する（ステップS52）。このリプレース対象の選択はLRU方式でよい。

さらに、制御部38は、ウェイ・レジスタ371を参照して、選択されたウェイがアクティブであるか否かを判定し（S53）、アクティブでなければステップS52に戻り再度他のウェイの
15 キャッシュエントリーを選択する。制御部38は選択されたアクティブなウェイのキャッシュエントリーをリプレースする（S54）。

このように、制御部38は、ウェイ・レジスタ371に示されるインアクティブなウェイに対しては、リプレースを制限し、制
20 御部38は、アクティブなウェイに対しては、リプレースを制限することなくキャッシュメモリとしても全機能を制御する。ここでは、リプレースの制限はリプレースの禁止としている。

図6は、プロセッサ1におけるタスク切り替え処理を示すフローチャートである。タスク切り替え処理は時間の経過やイベント
25 の発生により起動される。同図においてプロセッサ1は、現在実行中のタスクのコンテキストをメモリ2中の例えばスタック領

域に退避し（ステップ S 6 1）、次に実行すべきタスクのコンテキストをスタック領域から復帰させる（ステップ S 6 2）。ここで、スタック領域は、図 7 に示すように、メモリ 2 に確保され、各タスクのコンテキストを記憶するための領域である。各タスク
5 のコンテキストは、プロセッサの汎用レジスタのデータや、種々の制御レジスタのデータを含み、加えて、本実施の形態ではウェイ・レジスタに格納されるアクティブウェイデータを含む。

このようにして、ウェイ・レジスタ 3 7 1 は、タスク切り替えに際して書き換えられるので、常に実行中のタスクに対応する
10 アクティブウェイデータを保持することになる。

以上説明してきたように、本実施の形態におけるキャッシュメモリによれば、プロセッサ 1 に実行されるタスクから見れば、キャッシュメモリへのリードおよびライトについてはアクティブなウェイに対してもインアクティブなウェイに対しても可能
15 あるが、ミスヒットした場合にリプレース対象となるウェイについてはアクティブなウェイに制限されることになる。

例えば、図 4 において同図左側をタスク 1 実行時、右側をタスク 2 実行時のアクティブウェイとする。タスクの実行が経過するにつれて、タスク 1 のキャッシュデータは次第にウェイ 2 ～ 3 に
20 格納されていき、タスク 2 のキャッシュデータは次第にウェイ 4 ～ 7 に格納されていくことになる。言い換えれば、ウェイ 2 ～ 3 に格納されたタスク 1 のキャッシュデータはタスク 2 の実行によって追い出されない（リプレースされない）。また、ウェイ 4 ～ 7 に格納されたタスク 2 のキャッシュデータはタスク 1 の
25 実行によって追い出されない（リプレースされない）。その結果、タスク切り替えに伴ってタスク 1 では必要なキャッシュデータ

が、他のタスクによってリプレースされ、再度タスク 1 実行時に
追い出されたデータをキャッシュにリプレースすることも解消
できる。その結果、タスク切り替えに伴うリプレースの発生を低
減させることができ、他のタスクからの影響を抑えることができ
る。

＜変形例＞

なお、本発明のキャッシュメモリは、上記の実施形態の構成に
限るものではなく、種々の変形が可能である。以下、変形例のい
くつかについて説明する。

（１）複数のタスクと複数のウェイの対応関係については、１つ
のタスクに独占されるウェイと、複数のタスクに共用されるウェ
イとを混在させることができる。例えば、図 4 において、ウェイ
2～4 はタスク 1 が独占し、ウェイ 5～7 はタスク 2 が独占し、
ウェイ 0、1 は他のタスクが共用するものとする。この場合、タ
スク 1 および 2 は、ウェイを独占するので、タスク切り替えによ
るキャッシュミスを低減し、リアルタイム性を要する処理に適し
ている。タスク 1 および 2 以外のタスクは、リアルタイム性を要
しない処理等に適している。

（２）上記実施の形態では、制御部 38 は、アクティブなウェイ
についてはキャッシュメモリの全機能を制御し、インアクティブ
なウェイについてはリプレースを禁止しているが、これに限らな
い。

例えば、制御部 38 は、ウェイ・レジスタ 371 に示されたイン
アクティブなウェイ対して、その状態の更新を禁止する構成と
してもよい。例えば、ウェイの状態を表すフラグ類の更新を禁止
することにより、インアクティブなウェイに対するタスク切り替

えによる影響を防止することができる。

また、制御部 38 は、ウェイ・レジスタ 371 に示されたイン
アクティブなウェイについて、そのアクセス順序を示す情報の更
新を禁止する構成としてもよい。これによれば、アクセス順序を
5 示す情報の更新を禁止するので、他タスクの実行によりリプレー
ス順が影響を受けることがなくなる。

あるいは、制御部 38 は、インアクティブなウェイについて、
全機能を禁止するようにしてもよい。この場合、インアクティブ
なウェイのタグ出力を禁止するよう出力イネーブル信号をディ
10 スエーブルにすればよい。こうすれば、インアクティブなウェイ
の消費電力を低減することができる。また、全機能を禁止する場
合には、各タスクがウェイを共用することなく独占するように、
タスクとウェイとを対応付けることが望ましい。こうすれば、メ
モリとキャッシュメモリとの間でデータに矛盾が生じることを
15 防止することができる。

また、制御部 38 は、リプレースの禁止に加えて、アクセスの
順番を示す LRU 用の順序データを更新しないように構成して
もよい。

(3) ウェイ・レジスタ 371 の内容が更新されたとき、LRU
20 方式で用いられるアクセス順序情報をリセットする構成として
もよい。

(4) また、制御部 38 は、リプレース禁止の代わりにリプレー
ス回数を制限する構成としてもよいし、ウェイ中の特定のキャッ
シュエントリーに対するリプレースを禁止し、その他のキャッシ
25 ュエントリーに対してはリプレースを行う構成としてもよい。

(5) 上記実施の形態では、8ウェイ・セット・アソシエイティ

ブのキャッシュメモリを例に説明したが、ウェイ数は、4ウェイでも16ウェイでもいくつでもよい。また、上記実施の形態では、セット数が16である例を説明したが、セット数はいくつでもよい。

- 5 (6) 上記実施の形態では、セット・アソシエイティブのキャッシュメモリを例に説明したが、フル・アソシエイティブ方式のキャッシュメモリであってもよい。フル・アソシエイティブ方式の場合、セットが1つのケースと考えることができる。

- 10 (7) 図4に示したアクティブウェイデータは、OSによってタスク毎に割り当てられるようにしてもよい。すなわち、OSは実行対象のタスクを生成するときに、そのタスクに対してアクティブにすべきウェイを割り当て、割り当てた結果からそのタスクのアクティブウェイデータを生成する。さらに、OSは、生成されたアクティブウェイデータを、図7に示すように当該タスクのコン
15 テキストデータの一部に設定すればよい。

- (8) 複数のタスクがメモリデータを共有する場合、それらのタスクには全く同じウェイを共有すべきである。例えば、タスクAとタスクBがメモリデータを共有する場合、タスクAにウェイ5、6、7が割り当てられていれば、OSは、タスクBにもウェイ5、
20 6、7を割り当てればよい。

(実施の形態2)

- 実施の形態1では、ウェイ・レジスタ371をタスク切り替えによって書き換える構成を説明したが、本実施の形態では、キャッシュメモリにおいてタスクを判別して判別結果に応じてウェイ・レジスタ371を書き換える構成について説明する。加えて、
25 実施の形態1ではリプレースアルゴリズムが周知のLRU方式

としたが、本実施の形態ではアクセス順序を示すデータの代わりに１ビットのフラグを用いる擬似的なLRU方式を行う構成について説明する。

図８は、本発明の実施の形態２におけるキャッシュメモリの構成を示すブロック図である。同図のキャッシュメモリは、図２の構成と比較して、ウェイ３１ａ～３１ｄの代わりにウェイ１３１ａ～１３１ｄを備える点と、制御部３８の代わりに制御部１３８を備える点とが異なっている。以下、同じ点は説明を省略して、異なる点を中心に説明する。

ウェイ１３１ａは、ウェイ３１ａと比べて、各キャッシュエントリー中に、使用フラグとニューフラグとが追加されている点異なる。図９に、キャッシュエントリーのビット構成を示す。１つのキャッシュエントリーは、バリッドフラグＶ、２１ビットのタグ、１２８バイトのラインデータ、使用フラグＵ、ニューフラグＮおよびダーティフラグＤを保持する。このうち、使用フラグＵは、そのキャッシュエントリーにアクセスがあったか否かを示し、ミスヒットによるリプレースに際してセット内の８つのキャッシュエントリーにおけるアクセス順序の代わりに用いられる。より正確には、使用フラグＵの１は、アクセスがあったことを、０はないことを意味する。セット内の８つの使用フラグは、全て１になれば、０にリセットされるので、セット内の８つのキャッシュエントリーにおける使用の有無を示す相対的な値である。別言すれば、使用フラグＵは、アクセスされた時期が古いか新しいか２つの相対的な状態を示す。つまり、使用フラグＵが１のキャッシュエントリーは、使用フラグが０のキャッシュエントリーよりも新しくアクセスされたことを意味する。また、ニューフラグ

Nは、リプレース直後（又はフィル直後）に初期値として1が設定され、当該キャッシュエントリへのアクセスがあったときに0にリセットされる。つまり、ニューフラグNの1は、当該キャッシュエントリがリプレース（又はフィル）されてから一度も
5 アクセスされていない、新しい状態であることを意味する。

制御部138は、制御部38と比べて、設定部372が追加された点と、使用フラグUおよびニューフラグNの設定および更新を行う点とが異なる。

設定部372は、プロセッサ1において実行されているタスク
10 を判別し、判別したタスクに対応するアクティブウェイデータをウェイ・レジスタ371に設定する。

<設定部の構成>

図10は、設定部372の構成例を示すブロック図である。同図のように、設定部372は、判別部100a～100dとアク
15 ティブウェイデータ保持部110a～110dとセレクタ111とを備える。

判別部100aは、スタートアドレス保持部101、エンドアドレス保持部102、比較器103、104、アンド回路105とを有し、実行中のタスクがタスク1であることを判別する。

20 スタートアドレス保持部101、エンドアドレス保持部102は、プロセッサ1から読み書き可能であり、メモリ2に格納されたタスク1のスタートアドレス、エンドアドレスをそれぞれ保持する。このスタートアドレスおよびエンドアドレスは、プロセッサ1によって予め書き込まれ、動的に変更可能である。

25 比較器103は、プロセッサ1から出力される命令フェッチアドレス（IFアドレス）とスタートアドレス保持部101から出

力されるスタートアドレスとを比較し、スタートアドレスよりも
I F アドレスの方が大きい場合に 1 を出力する。

比較器 1 0 4 は、プロセッサ 1 から出力される I F アドレスと
エンドアドレス保持部 1 0 2 から出力されるエンドアドレスと
5 を比較し、I F アドレスよりもエンドアドレスの方が大きい場合
に 1 を出力する。

アンド回路 1 0 5 は、比較器 1 0 3 および 1 0 4 の比較結果が
共に 1 の場合、すなわち、I F アドレスがタスク 1 の命令をフェ
ッチしている場合に、実行されているタスクがタスク 1 であるこ
10 とを示す。

判別部 1 0 0 b ~ 1 0 0 d についても同様であり、実行中のタ
スクがタスク 2 ~ 3 であるかを判別する。

アクティブウェイデータ保持部 1 1 0 a ~ 1 1 0 d は、プロセ
ッサ 1 から読み書き可能であり、判別部 1 0 0 a ~ 1 0 0 d に対
15 応するタスクのアクティブウェイデータを保持する。このアクテ
ィブウェイデータは、プロセッサ 1 によって予め書き込まれ、動
的に変更可能である。

セクタ 1 1 1 は、判別部 1 0 0 a ~ 1 0 0 d の判別結果に従
って、実行中のタスクに対応するアクティブウェイデータを選択
20 し、ウェイ・レジスタ 3 7 1 に出力する。これにより、ウェイ・
レジスタ 3 7 1 は、実行中のタスクに対応するアクティブウェイ
データを保持する。

<使用フラグの更新例>

図 1 1 は、制御部 1 3 8 による使用フラグ U の更新例を示す説
25 明図である。同図では、説明の便宜上 8 ウェイではなく 4 ウェイ
の場合について説明する。同図の上段、中断、下段は、ウェイ 0

～ 3 に跨るセット N を構成する 4 つのキャッシュエントリーを示している。4 つのキャッシュエントリー右端の 1 又は 0 は、それぞれ使用フラグの値である。この 4 つの使用フラグ U を U 0 ～ U 3 と記す。

- 5 同図上段では (U 0 ～ U 3) = (1、0、1、0) であるので、ウェイ 0、2 のキャッシュエントリーはアクセスがあったことを、ウェイ 1、3 のキャッシュエントリーはアクセスがないことを意味する。

10 この状態で、メモリアクセスがセット N 内のウェイ 1 のキャッシュエントリーにヒットした場合、同図中断に示すように、(U 0 ～ U 3) = (1、1、1、0) に更新される。つまり、実線に示すようにウェイ 1 の使用フラグ U 1 が 0 から 1 に更新される。

15 さらに、同図中断の状態で、メモリアクセスがセット N 内のウェイ 3 のキャッシュエントリーにヒットした場合、同図下断に示すように、(U 0 ～ U 3) = (0、0、0、1) に更新される。つまり、実線に示すようにウェイ 3 の使用フラグ U 3 が 0 から 1 に更新される。加えて、破線に示すようにウェイ 3 以外の使用フラグ U 0 ～ U 2 が 1 から 0 に更新される。これにより、ウェイ 3 のキャッシュエントリーが、ウェイ 0 ～ 2 の各キャッシュエントリーよりも新しくアクセスされたことを意味することになる。

20 制御部 138 は、キャッシュミス時に使用フラグに基づいてリプレース対象のキャッシュエントリーを決定してリプレースを行う。例えば、制御部 138 は、図 11 上段では、ウェイ 1 とウェイ 3 の何れかをリプレース対象と決定し、図 11 中断ではウェイ 3 をリプレース対象と決定し、図 11 下段ではウェイ 0 ～ 2 の何れかをリプレース対象と決定する。

＜使用フラグ、ニューフラグの更新処理＞

図 1 2 は、制御部 1 3 8 における使用フラグおよびニューフラグのフラグ更新処理を示すフローチャートである。同図では、バリッドフラグが 0（無効）であるキャッシュエントリーの使用フラグ U は 0 に初期化されているものとする。

同図において、制御部 1 3 8 は、キャッシュヒットしたとき（ステップ S 6 1）、セットインデックスにより選択されたセットにおけるヒットしたウェイの使用フラグ U を 1 にセットし（ステップ S 6 2）、選択されたセット内のヒットしたウェイのキャッシュエントリーのニューフラグが 1 なら 0 にリセットする（ステップ S 1 7 1）。

さらに、制御部 1 3 8 は、そのセット内の他のウェイの使用フラグ U を読み出し（ステップ S 6 3）、読み出した使用フラグ U が全て 1 であるか否かを判定し（ステップ S 6 4）、全て 1 でなければ終了し、全て 1 であれば他のウェイの全ての使用フラグ U を 0 にリセットする（ステップ S 6 5）。

このようにして制御部 1 3 8 は、図 1 1 に示した更新例のように、使用フラグを更新する。また、ニューフラグ N は、キャッシュエントリーのリプレース後、最初にアクセスされた時点でリセットされる。

＜リプレース処理＞

図 1 3 は、制御部 1 3 8 におけるリプレース処理フローを示す図である。同図において制御部 1 3 8 は、メモリアクセスがミスしたとき（ステップ S 9 1）、セットインデックスにより選択されたセットにおける、8 つウェイの使用フラグ U と、8 つのニューフラグ N 0 ～ N 7 を読み出し（ステップ S 9 2）、読み出した

8つのニューフラグN 0 ~ N 7の全てが1であるか否かを判定し（ステップS 1 6 1）、全てが1である場合は、ステップS 9 3に進み、全てが1ではない（0がある）場合には、使用フラグUが0のウェイのうち、ニューフラグNが1のウェイを除外する（ステップS 1 6 2）。

さらに、制御部1 3 8は、使用フラグUが0のウェイを1つ選択する（ステップS 9 3）。このとき、使用フラグUが0になっているウェイが複数存在する場合は、制御部1 3 8はランダムに1つを選択する、あるいはラウンドロビン方式で1つを選択する。

さらに、制御部1 3 8は、当該セットにおける選択されたウェイのキャッシュエントリーを対象にリプレースし（ステップS 9 4）、リプレース後に当該キャッシュエントリーの使用フラグUを1に、ニューフラグを1に初期化する（ステップS 9 5）。なお、このときバリッドフラグV、ダーティフラグDは、それぞれ1、0に初期化される。また、ラウンドロビン方式で、使用フラグUが0になっている複数ウェイから1つのウェイを選択するために、制御部1 3 8をラウンド位置（選択したウェイの位置）を示すデータをレジスタに保持・更新し、使用フラグUが0になっているウェイのうち次のラウンド位置を選択すればよい。

このように、リプレース対象は、ニューフラグが0でかつ使用フラグが0のキャッシュエントリーを1つ選択することにより決定される。ただし、8つのニューフラグの全てが1である場合には、ニューフラグが1でかつ使用フラグUが0のウェイの中からリプレース対象を1つ選択する。このリプレースアルゴリズムは、従来のLRU方式におけるアクセス順序を示すデータの代わりに1ビットの使用フラグを用いるので、擬似的なLRU方式と

いうことができる。

以上説明してきたように、本実施の形態におけるキャッシュメモリによれば、設定部 372 を備えることにより、キャッシュメモリ自身が実行中のタスクを判別して、判別したタスクに対応するアクティブウェイデータをウェイ・レジスタ 371 に設定し、タスク毎にアクティブなウェイを切り替えることができる。その結果、実施の形態 1 と同様に、タスク切り替えに伴う無駄なリプレースの発生を低減させることができ、ヒット率を向上させることができる。

また、本実施の形態におけるキャッシュメモリによれば、従来の LRU 方式におけるアクセス順序を示すデータをキャッシュエントリー毎に設ける代わりに、1 ビットの使用フラグをキャッシュエントリー毎に設けている。これにより、従来のアクセス順序データを更新する複雑な回路を、使用フラグを更新する簡単なフラグ更新回路（フラグ更新部 39）に置き換えることができる。また、リプレース部 40 において、リプレース対象を、使用フラグが 0 のキャッシュエントリーの 1 つを選択することにより簡単に決定することができる。このように、本実施の形態におけるキャッシュメモリによれば、ハードウェア規模を大きく低減することができる。しかも、従来の LRU と比較してもほぼ同等のヒット率を得ることができる。

さらに、本実施の形態における制御部 138 は、ニューフラグが 1 の場合は、当該キャッシュエントリーをリプレース対象から除外している。これは、次の理由による。すなわち、使用フラグ U は初期値が 1 であるが他のウェイの使用フラグが順次 1 になれば、0 にリセットされる。つまり、使用フラグ U が 0 のキャッ

シュエントリーであってもリブレース後に一度もアクセスされていない場合がある。こうして使用フラグが0になった場合、リブレース後に一度もアクセスされていないキャッシュエントリーが、キャッシュミスの発生により再度リブレース対象に選択されてしまう可能性がある。そのため、ニューフラグNを設けることにより、リブレースされた後に一度もアクセスされていないキャッシュエントリーがリブレースされてしまうことを防止することができる。

＜変形例＞

10 なお、本発明のキャッシュメモリは、上記の実施の形態の構成に限るものではなく、種々の変形が可能である。以下、変形例のいくつかについて説明する。

（１）実施の形態１における変形例（１）～（６）を本実施の形態に適用してもよい。

15 （２）プロセッサ１から実行中のタスクを示すタスク番号（あるいはスレッド番号、プロセス番号等）が出力される場合には、上記判別部１００a～１００dの代わりに、タスク番号を保持および更新するタスク番号保持部を備える構成としてもよい。この場合セクタ１１１は、タスク番号に対応するアクティブウェイデータを選択すればよい。

20 （３）制御部１３８は、図１１の下段に示したようにセット内の他のウェイの使用フラグUが全部１であれば０にし、ヒットしたウェイ自身の使用フラグUを１に更新するが、この代わりに、ヒットしたウェイ自身の使用フラグも０に更新する構成としてもよい。

25 （４）ウェイ・レジスタ３７１の内容が更新されたとき、制御部

1 3 8 は、全ての使用フラグをリセットする構成としてもよい。
さらに、使用フラグのリセットと共に、制御部 1 3 8 は、使用フ
ラグ U が 0 になっている複数のウェイから 1 つを選択するため
の上記ラウンドロビン方式におけるラウンド位置を示す情報を
5 リセットしてもよい。

(5) 上記実施形態におけるニューフラグを有しない構成として
もよい。

(実施の形態 3)

実施の形態 1、2 では、キャッシュエントリーのリプレース単
10 位がライン (1 2 8 バイト) 単位でなされる構成を開示したが、
本実施の形態では、リプレース単位がタスク毎にライン単位とサ
ブライン (3 2 バイト) 単位とで切り替え可能な構成について説
明する。

図 1 4 は、本発明の実施の形態 3 におけるキャッシュメモリの
15 構成を示すブロック図である。同図のキャッシュメモリは、図 8
に示した構成と比較して、ウェイ 1 3 1 a ~ 1 3 1 h の代わりに
ウェイ 2 3 1 a ~ 2 3 1 h を備える点と、セレクタ 2 3 3 a ~ 2
3 3 h が追加された点と、制御部 1 3 8 の代わりに制御部 2 3 8
を備える点とが異なっている。以下、同じ点は説明を省略して異
20 なる点を中心に説明する。

ウェイ 2 3 1 a ~ 2 3 1 h は、図 8 のウェイ 1 3 1 a ~ 1 3 1
h と比べて、キャッシュエントリー内にバリッドフラグとダーテ
ィフラグとを 1 ビットずつ保持するのではなく、サブライン毎に
保持する点が異なっている。図 1 5 に、キャッシュエントリーの
25 ビット構成を示す。同図のように、同図のように 1 つのキャッシ
ュエントリーは、バリッドフラグ V 0 ~ V 3、タグ、ラインデー

タ、使用フラグU、ニューフラグN、ダーティフラグD0～D3を保持する。使用フラグUおよびニューフラグについては既に説明したので省略する。ラインデータ（128バイト）は4つのサブライン（32バイト）からなる。バリッドフラグV0～V3は、
5 4つのサブライン0～3に対応し、対応するサブラインが有効か否かを示す。ダーティフラグD0～D3は、4つのサブライン0～3に対応し、対応するサブラインに書き込みがあったか否かを示す。バリッドフラグおよびダーティフラグがサブライン単位に
10 設けられているのは、リプレースをサブライン単位でも行うことを可能にするためである。また、ライトバック（又はライトスルー）もサブライン単位で行うことが可能である。

セクタ233aは、ウェイ231aから、ソースインデックスS1により選択されたセットに対応するバリッドフラグV0～V3と、ワードインデックスW1の上位2ビットとが入力され、
15 この上位2ビットに指定されるサブラインに対応するバリッドフラグを選択する。セクタ233b～233hについても、ウェイ231b～231hに対応している点以外同様である。これによりセクタ233a～233hは、サブライン単位でヒットしたか否かを判定すること可能にしている。

20 制御部238は、制御部138と比べて、設定部372が削除された点と、RS（リプレースサイズ）レジスタ373が追加された点とが異なる。

設定部372が削除されているのは、ウェイ・レジスタ371が実施の形態1と同様にタスク切り替えにおいて書き換えられるからである。
25

RSレジスタ373は、ウェイ毎にリプレースサイズを示すリ

プレースサイズデータを保持する。図 1 6 に R S レジスタ 3 7 3 のビット構成例を示す。同図のように R S レジスタ 3 7 3 は、R S 0 ~ R S 7 からなるリプレースサイズデータを保持する。R S 0 ~ R S 7 の各ビットは、1 のときリプレースサイズがサブライ
5 ン（3 2 バイト）であることを、0 のときリプレースサイズがライン（1 2 8 バイト）であることを制御部 2 3 8 に指示する。この R S レジスタ 3 7 3 は、ウェイ・レジスタ 3 7 1 と同様に、プロセッサ 1 から読み書き可能であり、コンテキストの一部として
10 タスク切り換えにおいて書き換えられる。これにより、リプレースサイズをラインとするとサブラインとするかを、ウェイ毎にか
つ

タスク毎に設定することを可能にしている。

図 1 7 は、制御部 2 3 8 におけるフラグの更新処理を示すフローチャートである。同図は、図 1 2 に示したフローチャートと比
15 べて、ダーティフラグをサブライン単位で設定するためのステップ S 1 7 5 ~ S 1 7 7 が追加された点が異なる。すなわち、制御部 2 3 8 は、キャッシュメモリへの書き込みがあったとき（ステップ S 1 7 5）、書き込まれたサブラインを判別し（ステップ S 1 7 6）、判別されたサブラインに対応するダーティフラグを 1
20 にセットする（ステップ S 1 7 7）。ステップ S 1 7 5 ~ S 1 7 7 の処理は、例えば、制御部 2 3 8 に 1 入力 4 出力のデマルチプレクサをウェイ毎に備えることにより簡単に実現することができる。このデマルチプレクサは、論理” 1 ”が入力され、4 つの出力をキャッシュエントリー中のダーティフラグ D 0 ~ D 3 に
25 対応させ、ワードインデックス W I の上位 2 ビットにより出力先を制御するよう構成すればよい。

このようにして、制御部 238 は、サブライン単位に設けられたダーティフラグ D0～D3 をキャッシュライトに応じて更新する。

図 18 は、制御部 238 におけるリブレース処理を示すフローチャートである。同図は、図 13 に示したフローチャートと比べて、ステップ S94 の代わりにステップ S181～183 を有する点と、ステップ S95 の代わりにステップ S95a を有する点とが異なる。

制御部 238 は、RS レジスタ 373 からステップ S93 で選択されたウェイに対応する RS フラグを読み出して、リブレースサイズとしてサブラインとラインの何れが指定されているかを判定し（ステップ S181）、サブラインと指定されている場合には、当該ウェイのサブラインをリブレースし（ステップ S182）、ラインと指定されている場合には、当該ウェイのラインをリブレースする（ステップ S182）。さらに、制御部 238 は、リブレースされたサブラインまたはラインに対応するバリッドフラグおよびダーティフラグを初期化する（ステップ S95a）。すなわち、リブレースされたサブラインに対応するバリッドフラグ、ダーティフラグをそれぞれ 1、0 に設定する。ライン単位でリブレースされた場合には、4 つのサブラインに対応する 4 つのバリッドフラグ、ダーティフラグをそれぞれ 1、0 に設定する。

以上説明してきたように本実施の形態におけるキャッシュメモリによれば、実施の形態 1 又は 2 に加えて、リブレース単位をラインとサブラインとでウェイ毎およびタスク毎に設定できるので、タスクの必要とするデータサイズに応じてリブレース単位を切り換えることにより、キャッシュミスをもさらに低減すること

ができる。例えば、タスク 1 はオーディオデータのデコード／エンコード処理を、タスク 2 がビデオデータのデコード／エンコード処理を行うものとする。この場合、タスク 1 ではラインサイズをリプレース単位とし、タスク 2 ではサブラインをリプレース単位とすることができる。こうすれば、タスク 1、2 のキャッシュ利用効率を向上させることができる。なぜなら、タスク 1 は、シーケンシャルアクセスするデータの長さが比較的長く、タスク 2 は、シーケンシャルアクセスするデータの長さが比較的短いからである。

10 <変形例>

なお、本発明のキャッシュメモリは、上記の実施の形態の構成に限るものではなく、種々の変形が可能である。以下、変形例のいくつかについて説明する。

(1) 実施の形態 2 における変形例 (1)、(3)、(4)、(5) を本実施の形態に適用してもよい。

(2) 上記実施の形態では、サブラインのサイズをラインのサイズの 1/4 としているが、1/2、1/8、1/16 等他のサイズでもよい。その場合、各キャッシュエントリーは、サブラインと同数のバリッドフラグおよびダーティフラグをそれぞれ保持すればよい。

(実施の形態 4)

実施の形態 1～3 では、インアクティブなウェイに対して少なくともリプレースが制限される例を説明したが、本実施の形態では、さらに比較器 32a～32h のうちインアクティブなウェイに対応する比較器における比較の禁止と、インアクティブなウェイに対応するキャッシュエントリーからのタグ出力の禁止をす

る構成について説明する。

そのため本実施の形態におけるキャッシュメモリは図 1 4 に示した制御部 2 3 8 内部に比較制御部 3 7 2 を追加した構成となっている。

5 図 1 9 は、比較制御部 3 7 2 およびウェイ 2 3 1 a ~ 2 3 1 h の要部の構成を示すブロック図である。

同図において、キャッシュアドレスエントリ 3 0 0 a ~ 3 0 0 h は、キャッシュメモリ中のウェイ 0 ~ 8 に含まれ、それぞれセット数と同数（実施形態では 1 6 個）のタグを保持する。キャッシュアドレスエントリ 3 0 0 a は、ウェイ 0 に含まれ、1 6 個のタグ保持部 3 0 1 ~ 3 1 6 と、1 6 個のアンド回路 3 2 1 ~ 3 3 6 とを含む。他のキャッシュアドレスエントリ 3 0 0 b ~ 3 0 0 h も同様である。

15 アンド回路 3 2 1 は、1 6 個のセット中のセット 0 に対応し、セットインデックスをデコードするデコーダ 3 0 によりセット 0 が選択信号（s e t 0）と、イネーブル信号 E 0 とのアンドをとる。その結果、アンド回路 3 2 1 は、選択信号 s e t 0 = 1 かつイネーブル信号 E 0 = 1 のときのみ、タグ保持部 3 0 1 からのタグ出力と、比較器 3 2 a とをイネーブルにする。

20 比較器 3 2 a ~ 3 2 h は、それぞれイネーブル端子 E N を有し、イネーブル端子の入力が 1 のときに、アドレスレジスタ 2 0 中のタグアドレスとキャッシュアドレスエントリ 3 0 1 からのタグとの比較動作を行う。

25 比較制御部 3 7 2 は、8 つのウェイに対応する 8 つのイネーブル回路 3 8 1 a ~ 3 8 1 h、回数カウンタ 3 8 2 を備え、比較器 3 2 a ~ 3 2 h において最大 2 回のタグ比較を行わせるよう制

御し、1回目のタグ比較では、ウェイ・レジスタ371に示されたアクティブなウェイに対応する比較器をイネーブルにし、かつ
インアクティブウェイに対応する比較器をディスエーブルする
ことによって、アクティブウェイのタグを比較対象とし、さらに、

5 1回目のタグ比較においてミスヒットと判定された場合に、アク
ティブウェイに対応する比較器をディスエーブルし、アクティブ
ウェイに対応する比較器をイネーブルにすることによって、イン
アクティブウェイのタグを比較対象とし2回目のタグ比較を行
わせるよう構成している。また、各比較器のイネーブル／ディス
10 エーブルの制御と同時にウェイ231a～231hからのタグ
出力もイネーブル／ディスエーブルを制御している。これにより
ディスエーブルされた比較器およびタグ出力による消費電力の
低減を図っている。

イネーブル回路381a～381hは、ウェイ231a～23
15 1hに対応し、ウェイ・レジスタ371に保持されるアクティブ
ウェイデータに従って、比較器32a～32hのうち、1回目の
タグ比較ではアクティブなウェイに対応する比較器のみをイネ
ーブルにし、2回目のタグ比較ではインアクティブなウェイに対
応する比較器のみをイネーブルにする。

20 すなわち、イネーブル回路381a～381hは、イネーブル信
号E0～E7を生成し、このイネーブル信号E0～E7により、
キャッシュメモリ中のウェイ0～8に対応する8つのキャッシ
ュアドレスエントリーからのタグ出力と、8つの比較器32a～
32hをイネーブル／ディスエーブルする。例えば、イネーブル
25 回路381aは排他的論理和回路により、ウェイ0がアクティブ
か否かを示すW0ビットと、回数カウンタ382のカウント

値に従ってイネーブル信号 E_0 を生成する。

回数カウンタ 382 は、比較の回数をカウントするカウンタであり、0（1回目）、1（2回目）とカウントアップする。ただし、1回目がヒットした場合にはカウントアップしない。

- 5 図20は、回数カウンタ 382 のカウント値とアクティブウェイデータ W_n ($n = 0 \sim 7$) を入力として、イネーブル信号 E_n を出力とするイネーブル回路の制御論理を示す真理値表を示す。同図において、例えば、ウェイ 0～2 がアクティブでウェイ 3～7 がインアクティブである場合 ($W_0 = 1$)、1回目の比較では、
- 10 イネーブル信号 $E_0 \sim E_2$ が 1（イネーブル）でイネーブル信号 $E_3 \sim E_7$ がディスエーブルになる。この1回目の比較でヒットした場合には、回数カウンタがカウントアップしないので2回目の比較はなされない。1回目の比較でミスヒットした場合には、回数カウンタがカウントアップするので2回目の比較がなされ
- 15 る。この場合、2回目の比較では、イネーブル信号 $E_0 \sim E_2$ が 0（ディスエーブル）でイネーブル信号 $E_3 \sim E_7$ が 1（イネーブル）になる。

- 以上の構成により、1回目のタグ比較では、インアクティブなウェイを比較対照としないので比較器における消費電力および
- 20 タグ出力による消費電力を低減することができる。さらに、1回目のタグ比較においてミスヒットした場合には、インアクティブなウェイのみを比較対象として2回目のタグ比較を行うので、全てのウェイのキャッシュデータを有効に活用することができる。

- なお、2回目の比較において全ての比較器をイネーブルにしてもよい。この場合、1回目でのヒットした場合に2回目の比較が行われないので、消費電力を低減することができる。つまり、タ
- 25

スクごとにアクティブウェイが割り当てられるので1回目の比較において高いヒット率を得られると考えられる。

また、2回目の比較を行わない構成としてもよい。この1回目の比較において高いヒット率を得られると考えられるので、消費電力を低減する効果がある。

産業上の利用可能性

本発明は、メモリアクセスを高速化するためのキャッシュメモリおよびその制御方法に適しており、例えば、オンチップキャッシュメモリ、オフチップキャッシュメモリ、データキャッシュメモリ、命令キャッシュメモリ等に適している。

請 求 の 範 囲

1. N-ウェイ・セット・アソシエイティブ方式のキャッシュメモリであって、

- 5 N個のウェイのうち1つ以上のウェイを示す制御レジスタと、
制御レジスタに示されるウェイをアクティブにする制御手段と、

制御レジスタの内容を更新する更新手段と
を備えることを特徴とするキャッシュメモリ。

10

2. 前記制御手段は、制御レジスタに示されたアクティブなウェイ以外のウェイに対して少なくともリプレースを制限することを特徴とする請求項1記載のキャッシュメモリ。

- 15 3. 前記キャッシュメモリは、さらに、

ウェイ毎に設けられ、キャッシュデータのアドレスをタグとして保持するタグ保持手段と、

- プロセッサから出力されるメモリアクセスアドレスの上位部分であるタグアドレスと、タグ保持手段から出力されるN個のタグとを比較することによりヒットかミスヒットかを判定するN
20 個の比較手段を有し、

前記制御手段は、制御レジスタに示されたアクティブなウェイ以外のウェイに対応する比較手段をディスエーブルにする

ことを特徴とする請求項1記載のキャッシュメモリ。

25

4. 前記制御手段は、さらに、制御レジスタに示されたアクティ

ブなウェイ以外のウェイに対応するキャッシュアドレス保持手段に対して、比較手段へのタグ出力をディスエーブルにする

ことを特徴とする請求項 3 記載のキャッシュメモリ。

- 5 5. 前記制御手段は、プロセッサからメモリアクセスアドレスが出力されたとき、当該アクセスアドレスについて、比較手段に最大 2 回のタグ比較を行わせるよう制御し、

10 1 回目のタグ比較において制御レジスタに示されたアクティブなウェイ以外のウェイに対応する比較手段をディスエーブルし、

1 回目のタグ比較においてミスヒットと判定された場合に、アクティブなウェイ以外のウェイに対応する比較手段をディスエーブルしないで 2 回目のタグ比較を行わせる

ことを特徴とする請求項 3 記載のキャッシュメモリ。

15

6. 前記制御手段は、前記 2 回目のタグ比較においてアクティブなウェイに対応する比較手段をディスエーブルする

ことを特徴とする請求項 5 記載のキャッシュメモリ。

- 20 7. 前記制御手段は、制御レジスタに示されたアクティブなウェイ以外のウェイに対して、その状態の更新を禁止する

ことを特徴とする請求項 2 記載のキャッシュメモリ。

- 25 8. 前記制御手段は、制御レジスタに示されたアクティブなウェイ以外のウェイについて、そのアクセス順序を示す情報の更新を禁止する

ことを特徴とする請求項 2 記載のキャッシュメモリ。

9. 前記キャッシュメモリは、さらに

- 前記更新手段によって制御レジスタの内容が更新されたとき、
5 ウェイに対するアクセス順序を示す情報をリセットするリセット手段を有する

ことを特徴とする請求項 2 記載のキャッシュメモリ。

10. 前記アクセス順序を示す情報は、キャッシュエントリ毎
10 の 1 ビットデータであり、

前記キャッシュメモリは、さらに、リプレース可能な複数ウェイから 1 つのウェイをラウンドロビン方式で選択するためラウンド位置を示すデータを保持するレジスタを有し、

- 前記リセット手段は、前記更新手段によって制御レジスタの内容
15 が更新されたとき、前記レジスタをリセットする

ことを特徴とする請求項 9 記載のキャッシュメモリ。

11. 前記更新手段は、

- アクティブにすべきウェイを指定するウェイデータであって、
20 タスク毎のウェイデータを保持する保持手段と、

実行中のタスクに対応するウェイデータを保持するよう前記制御レジスタを書き換える書き換え手段と

を有することを特徴とする請求項 2 記載のキャッシュメモリ。

- 25 12. 前記保持手段は、メモリ中に記憶されたタスク毎のコンテキストデータの一部として前記ウェイデータを保持し、

前記書き換え手段は、タスク切り替えに際して、制御レジスタ中の現タスクのウェイデータをメモリに退避し、次タスクのウェイデータをメモリから前記制御レジスタに復帰する

ことを特徴とする請求項 11 記載のキャッシュメモリ。

5

13. 前記保持手段は、タスク毎の前記ウェイデータを保持し、前記書き換え手段は、

メモリに記憶された各タスクのアドレス範囲を記憶するアドレス記憶手段と、

10 アドレス記憶手段に記憶されたアドレス範囲と、プロセッサから出力される命令フェッチアドレスとに基づいて、実行中のタスクを判別する判別手段と、

判別された実行中のタスクに対応するウェイデータを前記保持手段から選択する選択手段と、

15 選択されたウェイデータを前記制御レジスタに書き込む書き込み手段と

を備えることを特徴とする請求項 12 記載のキャッシュメモリ。

20 14. 前記保持手段は、タスク毎の前記ウェイデータを保持し、前記書き換え手段は、

プロセッサから出力されるタスク番号に従って、実行中のタスクに対応するウェイデータを前記保持手段から選択する選択手段と、

25 選択されたウェイデータを前記制御レジスタに書き込む書き込み手段と

を備えることを特徴とする請求項 12 記載のキャッシュメモリ。

15 15. 前記保持手段に保持されるウェイデータは、オペレーティングシステムによってタスクに割り当てられることを特徴とする請求項 11 記載のキャッシュメモリ。

10 16. 前記キャッシュメモリは、各ウェイにおけるリプレース単位をキャッシュエントリーのラインサイズと、ラインサイズの 2 の n 乗分の 1 のサイズとに切り替え可能であり、

前記制御レジスタは、さらに、ウェイ毎のリプレースサイズを示し、

前記制御手段は、制御レジスタに示されたリプレースサイズを単位としてリプレース制御を行う

15 ことを特徴とする請求項 1 記載のキャッシュメモリ。

17. 前記制御手段は、制御レジスタに示されたアクティブなウェイ以外のウェイに対して少なくともリプレースを制限し、制御レジスタに示されたアクティブなウェイに対して制御レジスタに示されたサイズを単位にリプレースを行う

ことを特徴とする請求項 16 記載のキャッシュメモリ。

18. 前記更新手段は、

25 アクティブにすべきウェイを指定するウェイデータであってタスク毎のウェイデータと、タスク毎のリプレースサイズとを保持する保持手段と、

実行中のタスクに対応するウェイデータ及びリプレースサイズを保持するよう前記制御レジスタを書き換える書き換え手段と
を有することを特徴とする請求項 17 記載のキャッシュメモリ。

5

19. 前記キャッシュメモリは、さらに、
キャッシュの単位となるデータを保持するキャッシュエントリー毎に、アクセスの有無を示す 1 ビットのアクセス情報を記憶する記憶手段と、

10 アクセス無しを示すアクセス情報に対応するキャッシュエントリーの中からリプレース対象のキャッシュエントリーを選択する選択手段と

を備えることを特徴とする請求項 1 記載のキャッシュメモリ。

15 20. 前記キャッシュメモリは、さらに

リプレース可能な複数ウェイから 1 つのウェイをラウンドロビン方式で選択するためのラウンド位置を示すデータを保持するレジスタを有し、

20 前記更新手段によって制御レジスタの内容が更新されたとき、
ウェイに対するアクセス順序を示す情報と、前記レジスタをラウンド位置を示すデータとをリセットするリセット手段と

を有することを特徴とする請求項 19 記載のキャッシュメモリ。

25 21. N-ウェイ・セット・アソシエイティブ方式のキャッシュメモリを制御する制御方法であって、

N 個のウェイのうち 1 つ以上のウェイを示すウェイデータを制御レジスタに設定するステップと、

制御レジスタに示されるウェイをアクティブにする制御ステップと

5 を有することを特徴とする制御方法。

2 2 . 前記制御ステップでは、制御レジスタに示されたアクティブなウェイ以外のウェイに対して少なくともリプレースを制限する

10 ことを特徴とする請求項 2 1 記載の制御方法。

2 3 . 前記制御方法は、さらに、

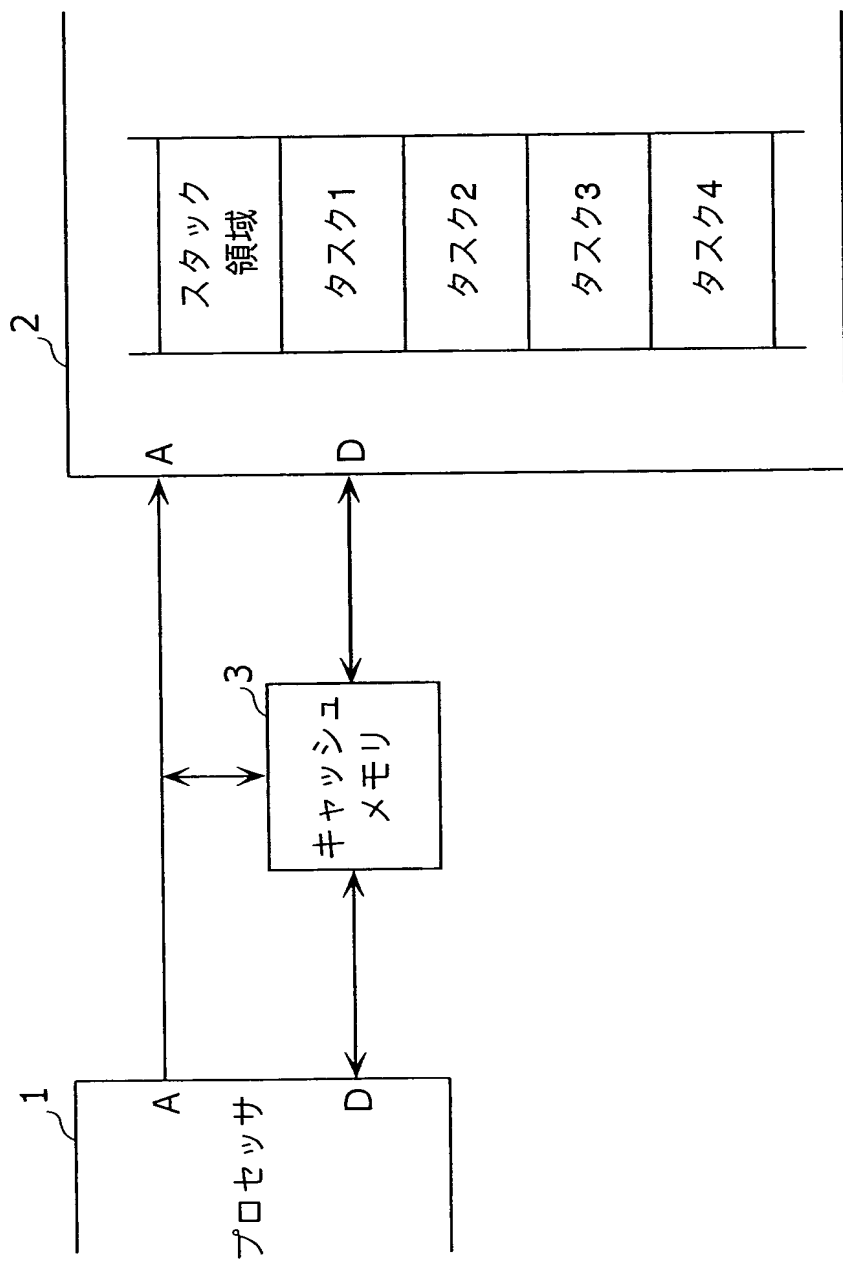
アクティブにすべきウェイを指定するウェイデータであって
タスク毎のウェイデータを保持する保持部から、実行中のタスク
15 に対応するウェイデータ読み出して、読み出したウェイデータを
前記制御レジスタに書き込む更新ステップを有する

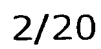
ことを特徴とする請求項 2 2 記載の制御方法。

要 約 書

本発明のキャッシュメモリは、N－ウェイ・セット・アソシエ
イティブ方式のキャッシュメモリであって、N個のウェイのうち
5 1つ以上のウェイを示す制御レジスタと、制御レジスタに示され
るウェイをアクティブにする制御手段と、制御レジスタの内容を
更新する更新手段とを備え、前記制御手段は、制御レジスタに示
されたアクティブなウェイ以外のウェイに対して少なくともリ
プレースを制限する。

図1





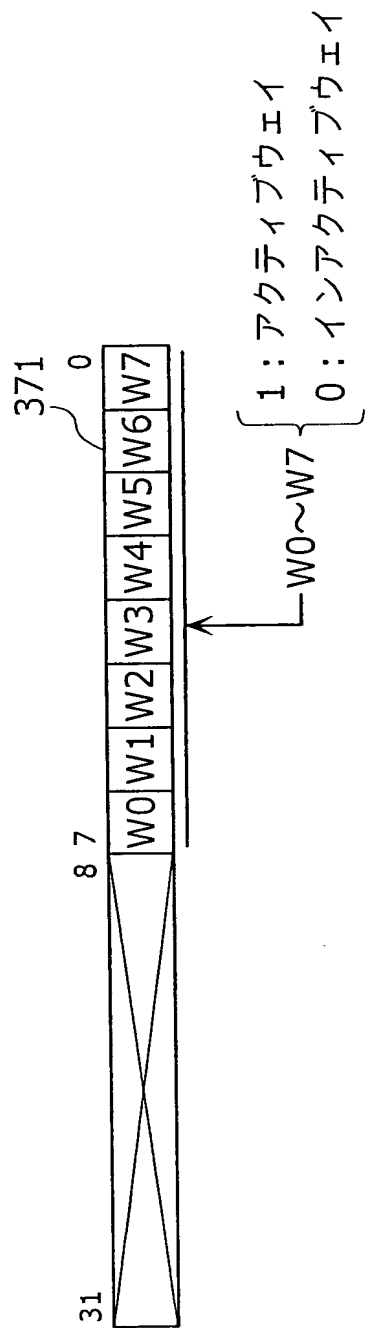


図3

図4

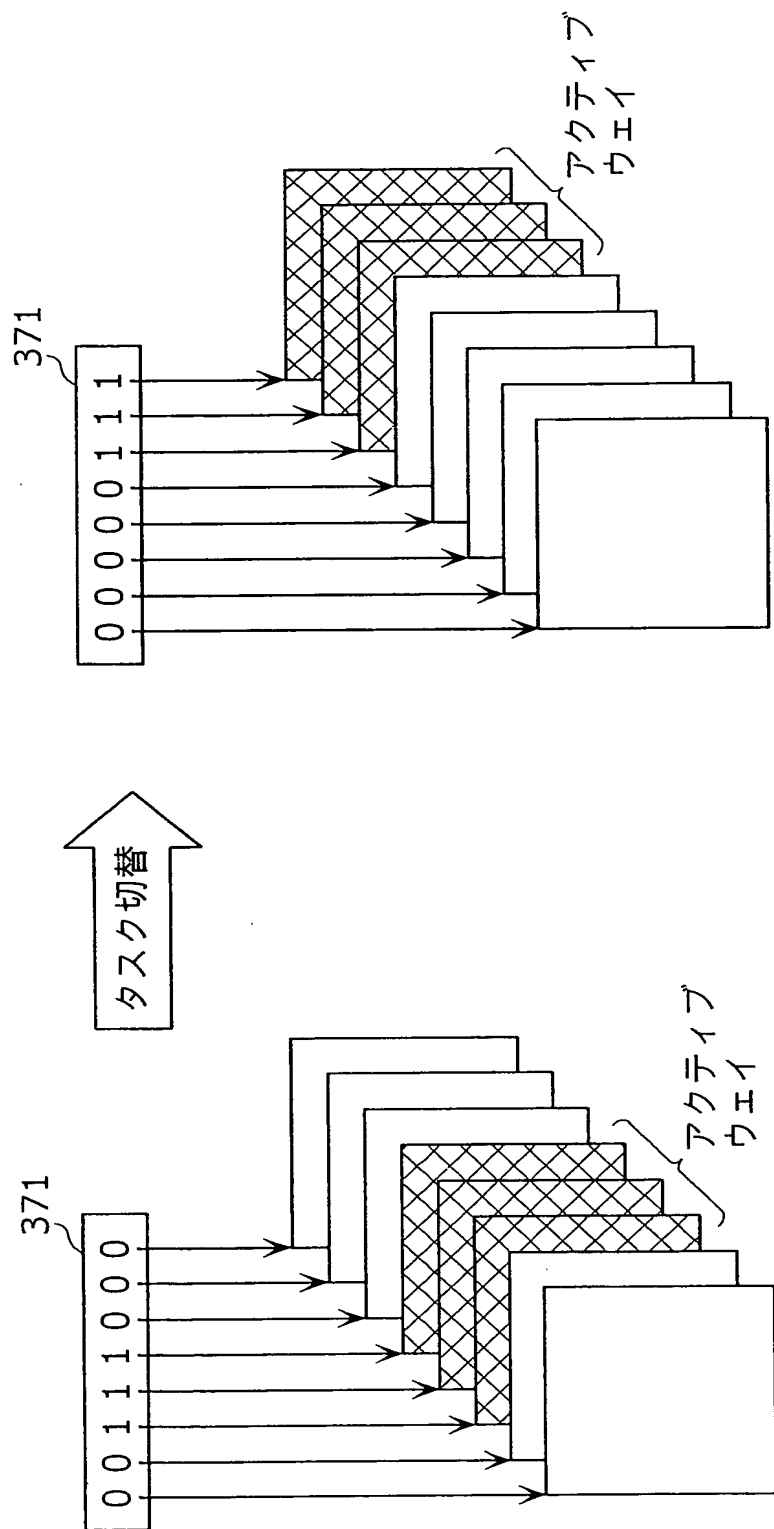


図5

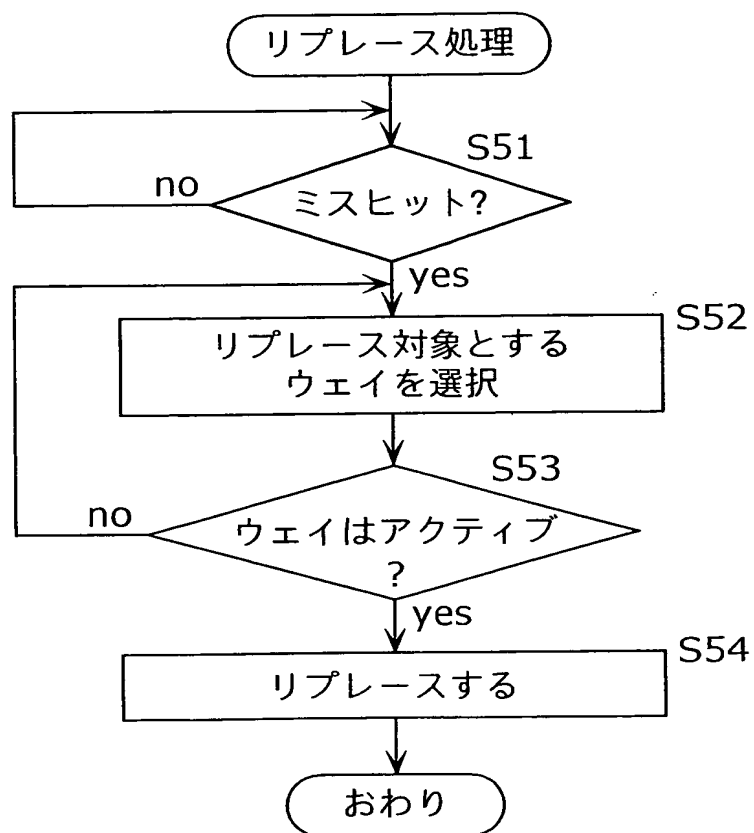


図6

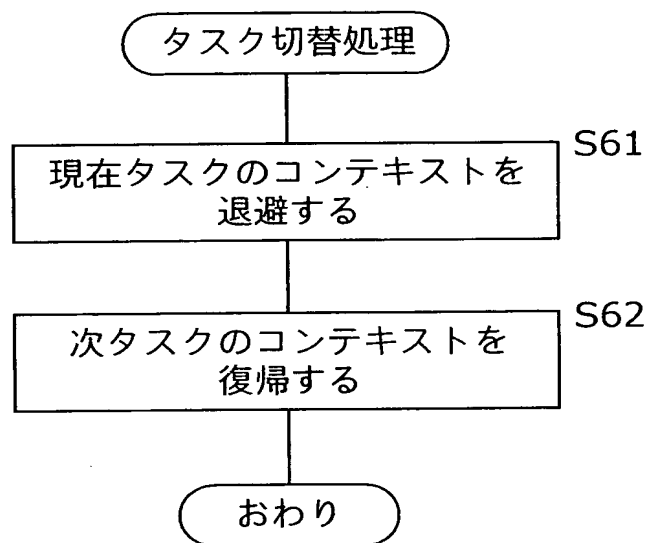
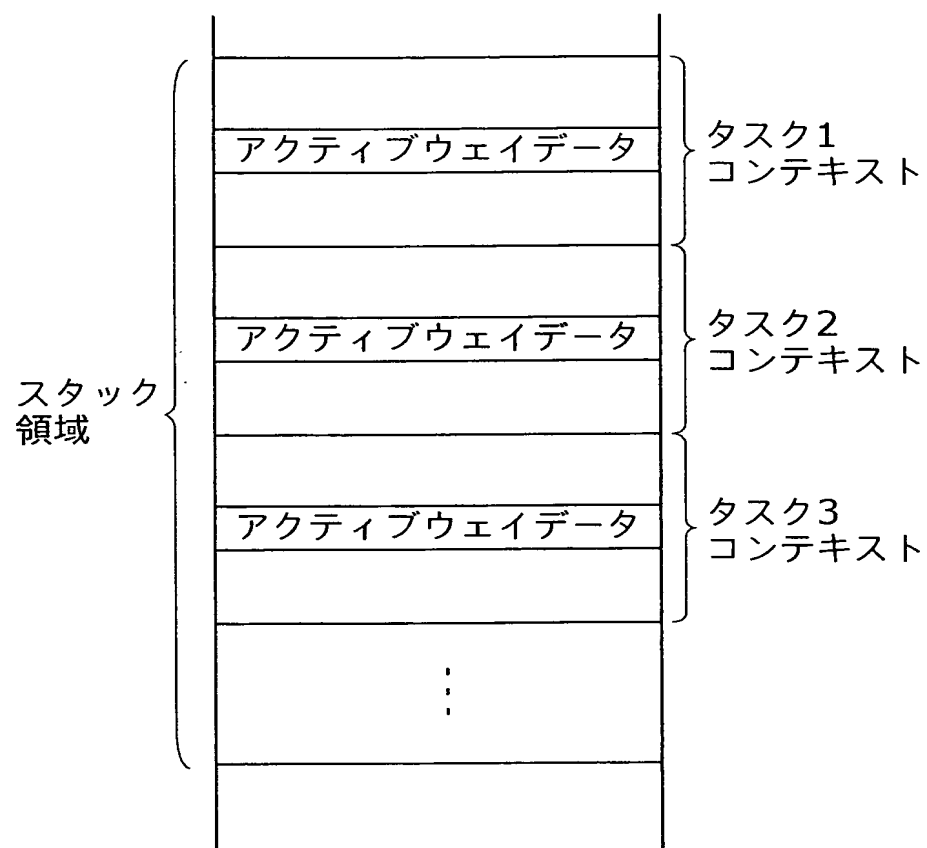


図7



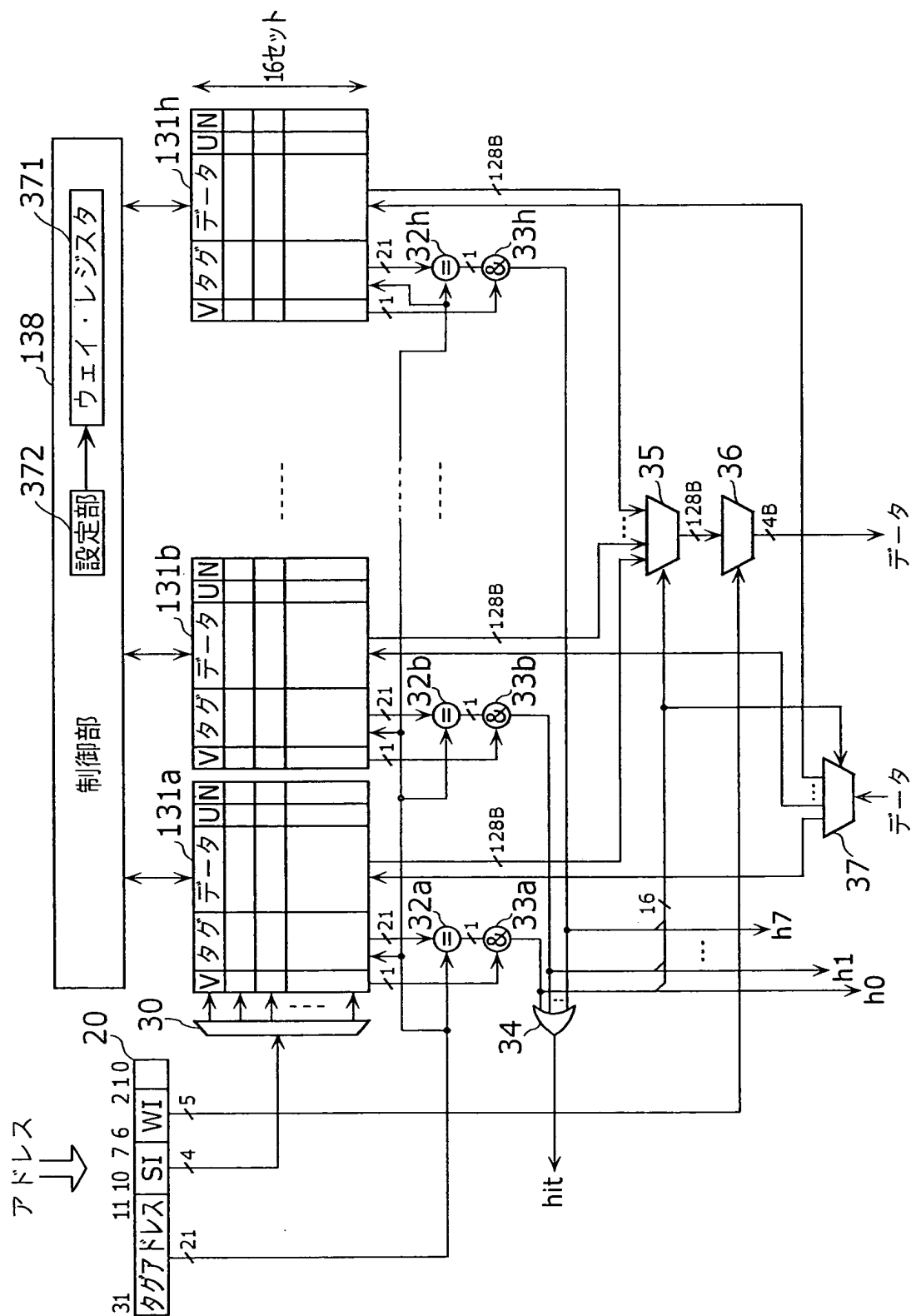


図9

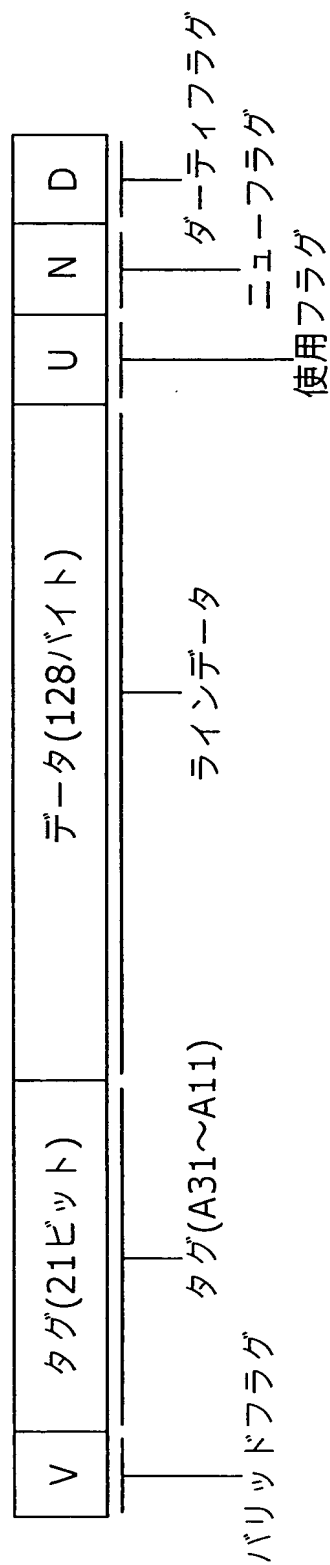
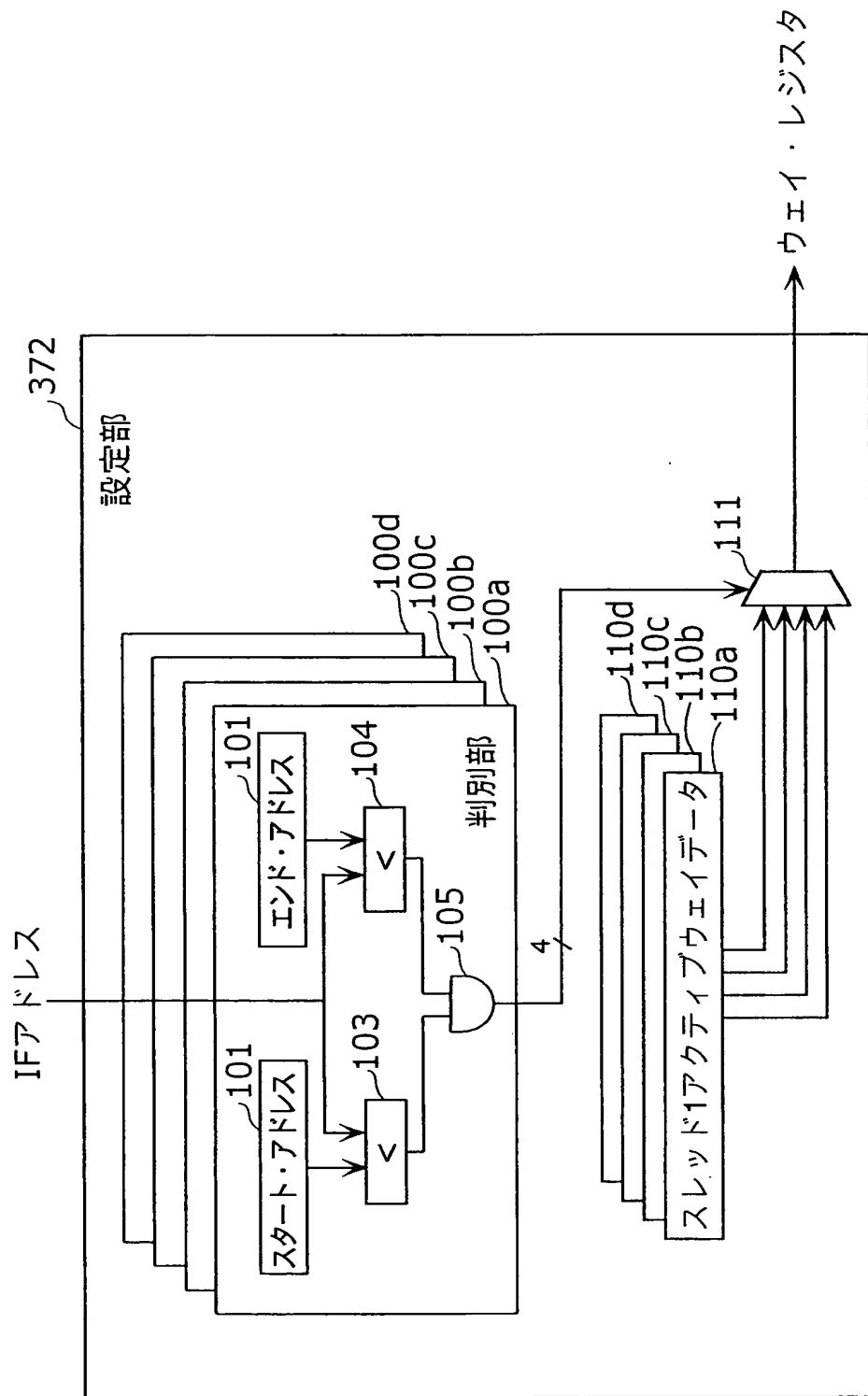


図10



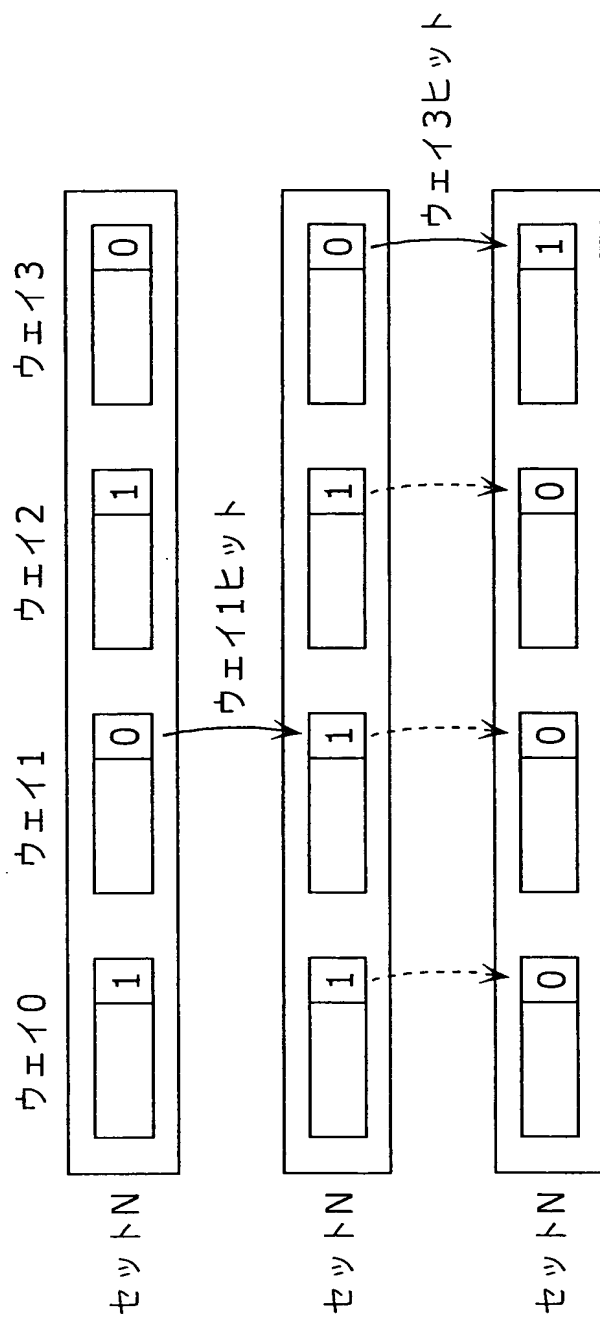


図11

図12

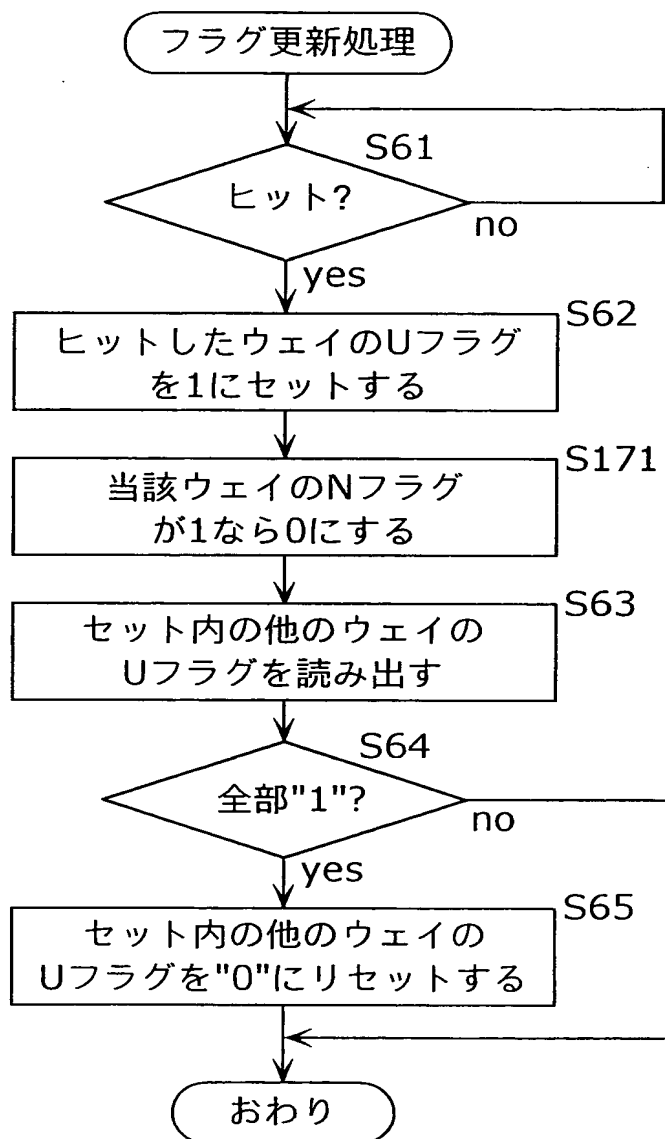


図13

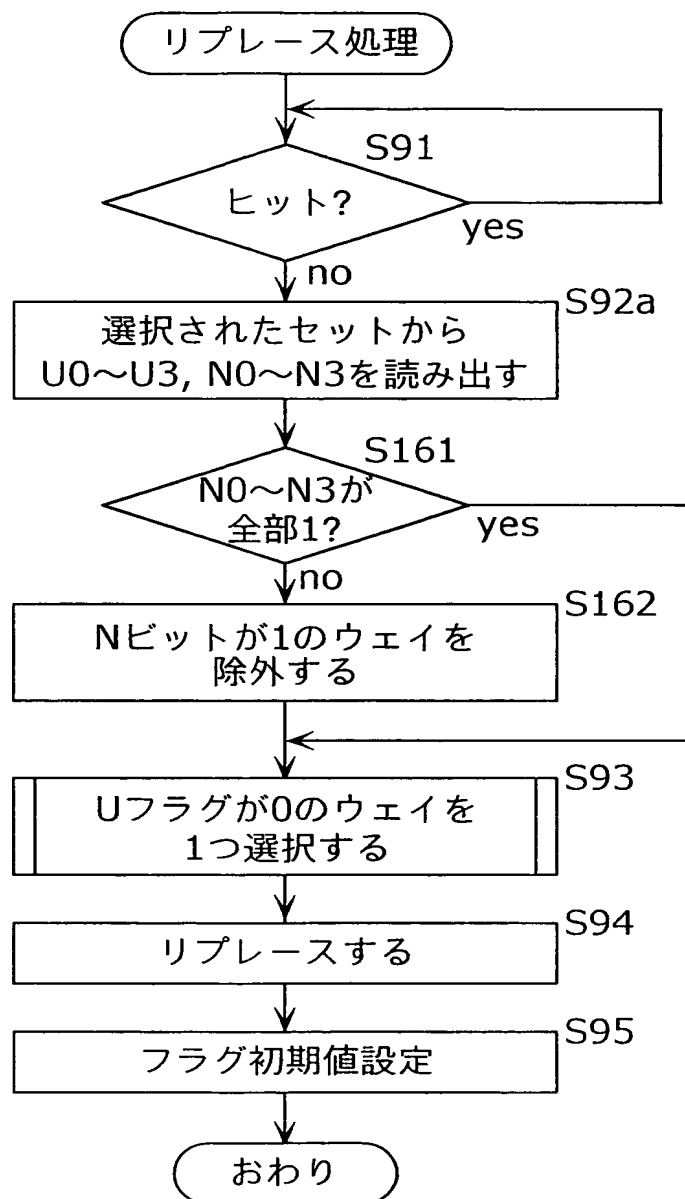


図15

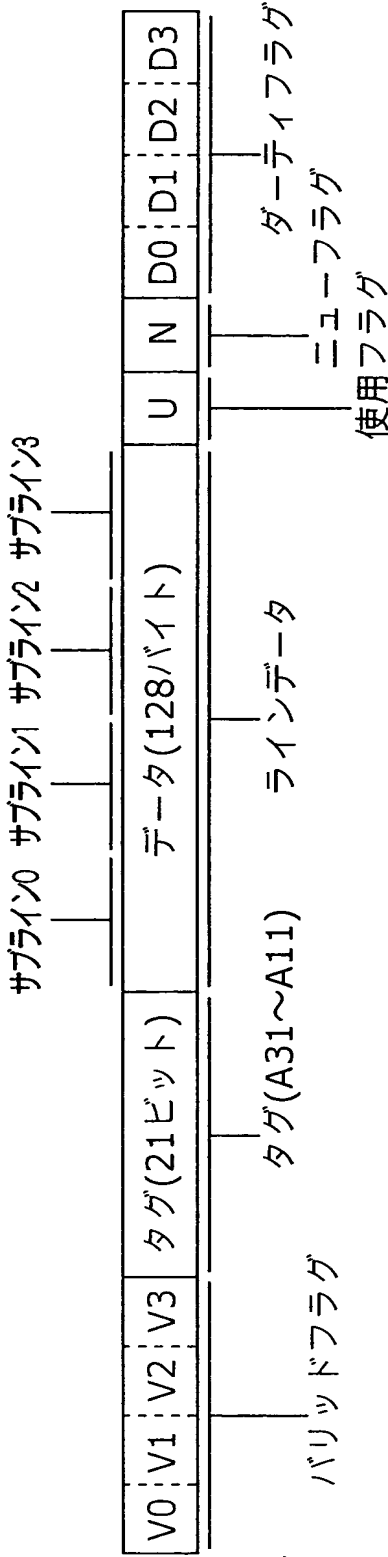


図16

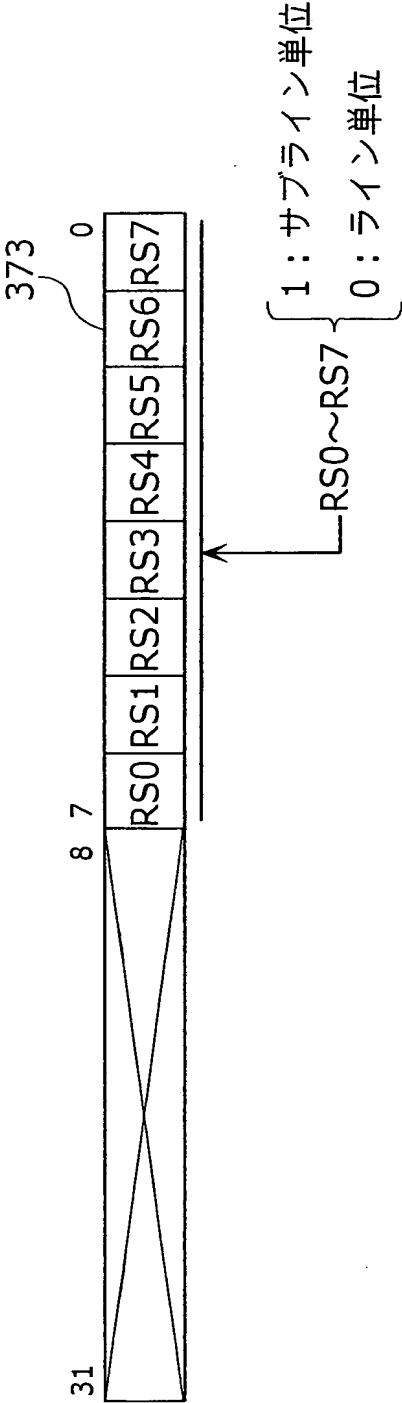


図17

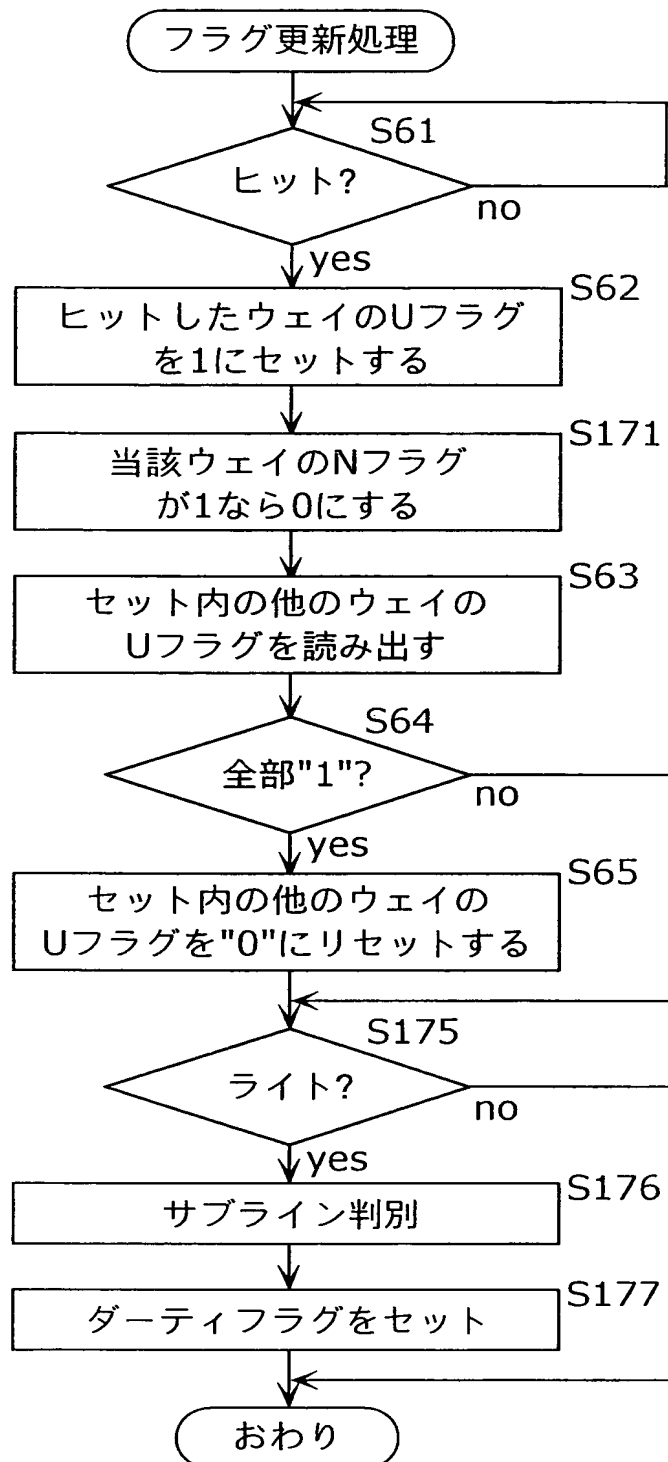


図18

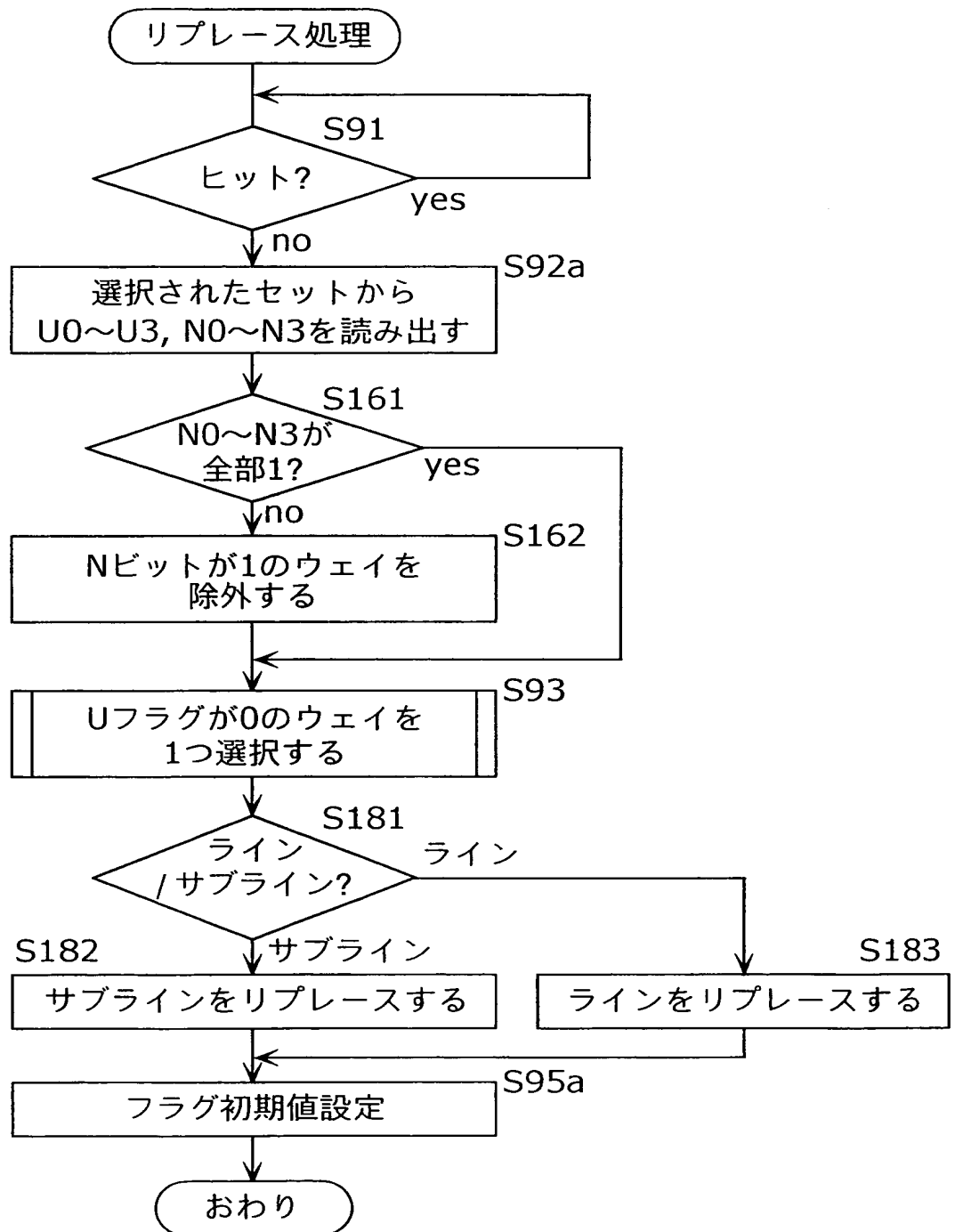


図19

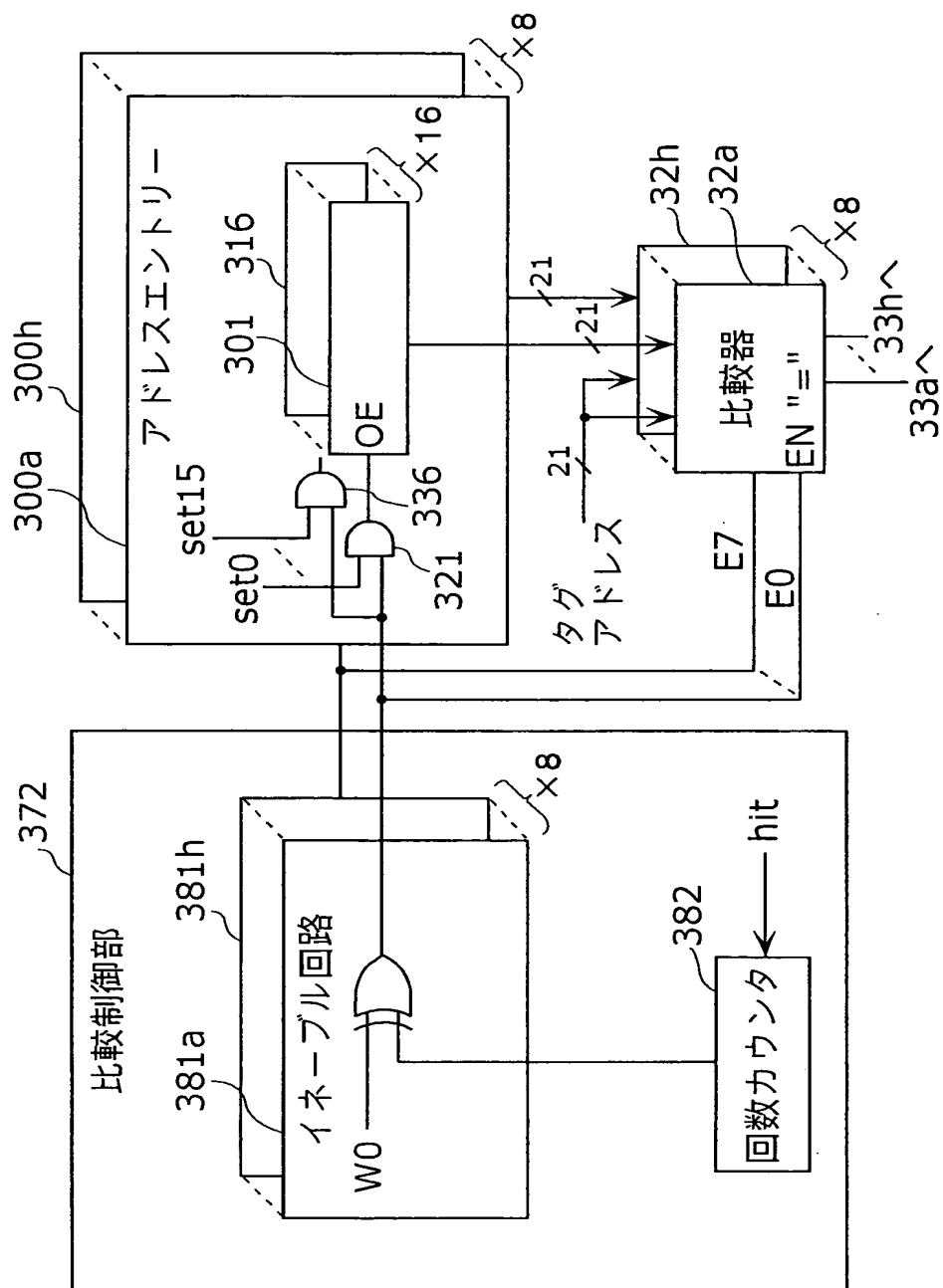


図20

入力		出力
Wn(n=0~7)	カウント値	En
1(アクティブ)	0(1回目)	1(イネーブル)
1(アクティブ)	1(2回目)	0(ディスエーブル)
0(インアクティブ)	0(1回目)	0(ディスエーブル)
0(インアクティブ)	1(2回目)	1(イネーブル)